PCT WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification 6 : G06F 17/17, 17/27 — A1

(11) International Publication Number: WO 00/05658

(43) International Publication Date: 3 February 2000 (03.02.00)

(21) International Application Number: PCT/US99/16654

(22) International Filing Date: 23 July 1999 (23.07.99)

(30) Priority Data:

60/094,051	24 July 1998 (24.07.98)	US
60/094,079	24 July 1998 (24.07.98)	US
60/094,066	24 July 1998 (24.07.98)	US

(71) Applicant *(for all designated States except US)*: OMNI-VISTA, INC. [US/US]; 4350 ArrowsWest Drive, Colorado Springs, CO 80907 (US).

(72) Inventors; and

(75) Inventors/Applicants *(for US only)*: DAVIS, Alan, M. [US/US]; 770 Nebula Court, Colorado Springs, CO 80906 (US). ZWEIG, Ann, S. [US/US]; 6154 Colony Circle, Colorado Springs, CO 80919 (US).

(74) Agent: HALLING, Dale, B.; Suite 202, 128 S. Tejon, Colorado Springs, CO 80903 (US).

(81) Designated States: AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, CA, CH, CN, CU, CZ, DE, DK, EE, ES, FI, GB, GE, GH, HU, IL, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MD, MG, MK, MN, MW, MX, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, UA, UG, US, UZ, VN, YU, ZW, ARIPO patent (GH, GM, KE, LS, MW, SD, SL, SZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG).

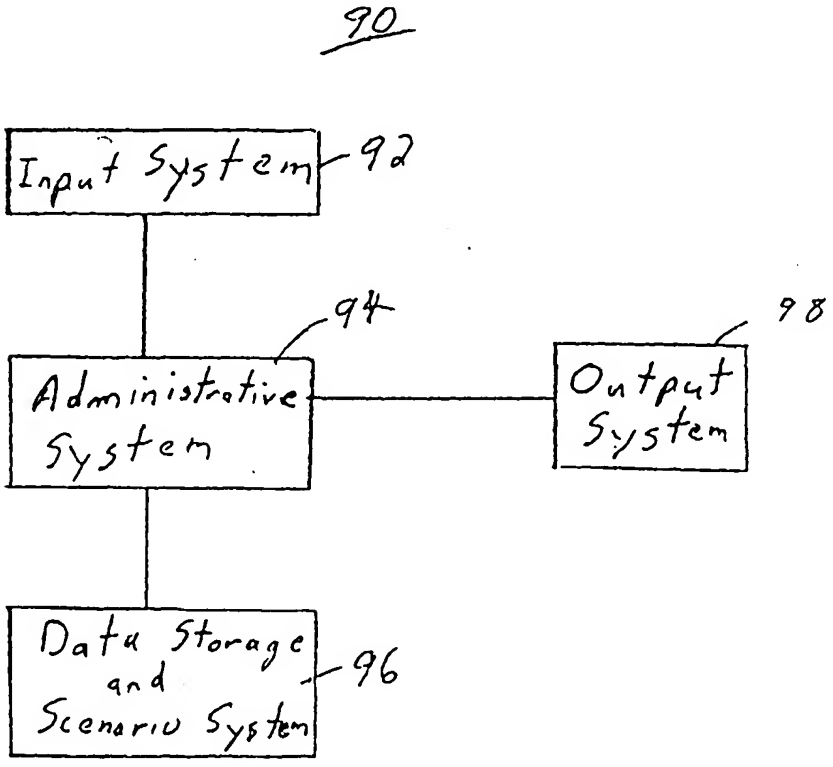
Published

With international search report.

(54) Title: METHOD FOR PROVIDING BI-DIRECTIONAL PROPAGATION AMONG DATA WITHIN SPREADSHEETS

(57) Abstract

The analysis package (90) includes an input system (92) for receiving data. An administrative system (94) receives the data from the input system (92). A data storage and scenario system (96) receives the data from the administrative system (94) and forms a scenario case. An output system (98) is connected to the administrative system (94) and displays the scenario case.



FOR THE PURPOSES OF INFORMATION ONLY

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

AL	Albania	ES	Spain	LS	Lesotho	SI	Slovenia
AM	Armenia	FI	Finland	LT	Lithuania	SK	Slovakia
AT	Austria	FR	France	LU	Luxembourg	SN	Senegal
AU	Australia	GA	Gabon	LV	Latvia	SZ	Swaziland
AZ	Azerbaijan	GB	United Kingdom	MC	Monaco	TD	Chad
BA	Bosnia and Herzegovina	GE	Georgia	MD	Republic of Moldova	TG	Togo
BB	Barbados	GH	Ghana	MG	Madagascar	TJ	Tajikistan
BE	Belgium	GN	Guinea	MK	The former Yugoslav Republic of Macedonia	TM	Turkmenistan
BF	Burkina Faso	GR	Greece			TR	Turkey
BG	Bulgaria	HU	Hungary	ML	Mali	TT	Trinidad and Tobago
BJ	Benin	IE	Ireland	MN	Mongolia	UA	Ukraine
BR	Brazil	IL	Israel	MR	Mauritania	UG	Uganda
BY	Belarus	IS	Iceland	MW	Malawi	US	United States of America
CA	Canada	IT	Italy	MX	Mexico	UZ	Uzbekistan
CF	Central African Republic	JP	Japan	NE	Niger	VN	Viet Nam
CG	Congo	KE	Kenya	NL	Netherlands	YU	Yugoslavia
CH	Switzerland	KG	Kyrgyzstan	NO	Norway	ZW	Zimbabwe
CI	Côte d'Ivoire	KP	Democratic People's Republic of Korea	NZ	New Zealand		
CM	Cameroon			PL	Poland		
CN	China	KR	Republic of Korea	PT	Portugal		
CU	Cuba	KZ	Kazakstan	RO	Romania		
CZ	Czech Republic	LC	Saint Lucia	RU	Russian Federation		
DE	Germany	LI	Liechtenstein	SD	Sudan		
DK	Denmark	LK	Sri Lanka	SE	Sweden		
EE	Estonia	LR	Liberia	SG	Singapore		

METHOD FOR PROVIDING BI-DIRECTIONAL PROPAGATION AMONG DATA WITHIN SPREADSHEETS

Field of the Invention

The present invention relates generally to the field of application software and more particularly to a method for providing bi-directional propagation among data within spreadsheets.

Background of the Invention

Spreadsheets are standard tools used on computers by individuals and businesses to help them with their decision making. Spreadsheets are composed of sets of two-dimensional arrays of cell. Each cell contains either a constant (i.e., a number) or a formula that enables the cell's value to change whenever any cell referred to by the formula is changed. Circular references are not allowed. Thus if cell A contains a formula that refers to cell B, then cell B cannot contain a formula that refers to cell A. Such circular reference cannot be 'resolved" by spreadsheets because it creates an infinite loop. In most spreadsheet applications this is acceptable. in fact, creating a circular reference in a spreadsheet is usually caused by an error in logic by the user. However, sophisticated application being built by sophisticated spreadsheet users have a need to create and use circular references.

Some software programs use goal seeking to resolve the circular reference problem. One method used by these software programs is linear programming, while another method is non-linear programming. For example. when cell G is referred to by cell F and the user tries to change the content of cell F, these programs attempt in real-time to turn the equation inside out, trying to solve the equation for G in terms of F. This is non-trivial, especially if F is a function of many other cell as well. As a result, the results have been unsatisfactory.

Thus there exists a need for a method for providing bi-directional propagation among data within spreadsheets that is easy to use and effective even for complex equations.

Brief Description of the Drawings

FIG. 1 is a front view of a computer capable of operating the invention in accordance with one embodiment of the invention;

FIG. 2 is a block diagram of the invention in accordance with one embodiment of the invention;

FIG. 3 is a block diagram of the invention in accordance with one embodiment of the invention;

FIG. 4 is a block diagram of an input system of the invention in accordance with one embodiment of the invention;

FIG. 5 is a schematic diagram of a data storage and scenario system of the invention in accordance with one embodiment of the invention;

FIG. 6 is a flow chart of the steps used in operating the invention in accordance with one embodiment of the invention;

FIGs. 7-12 are example screens of an interview system of the invention in accordance with one embodiment of the invention;

FIGs. 13-14 are example screens of an output system of the invention in accordance with one embodiment of the invention;

FIG. 15 is an example screen of an output system of the invention in accordance with one embodiment of the invention;

FIG. 16 is a flow chart of the steps used in resolving circular references in accordance with one embodiment of the invention;

FIG. 17 is a flow chart of the steps used in resolving circular references in accordance with one embodiment of the invention;

FIG. 18 is a list of rules used in resolving a circular reference in accordance with one embodiment of the invention;

FIG. 19 is a flow chart of the steps used in implementing the rules to resolve a circular reference in accordance with one embodiment of the invention;

FIG. 20 is a front view of a computer capable of operating the invention in accordance with one embodiment of the invention;

FIG. 21 is a block diagram of the invention in accordance with one embodiment of the invention;

FIG. 22 is a flow chart of the steps used in the invention in accordance with one embodiment of the invention;

FIGs. 23A-23B are example screens of an output system of the invention in accordance with one embodiment of the invention;

FIG. 24 is an example screen of an interview system of the invention in accordance with one embodiment of the invention;

FIG. 25 is an example screen of an output system of the invention in accordance with one embodiment of the invention;

FIG. 26 is a flow chart of the steps used in the invention in accordance with one embodiment of the invention;

FIGs. 27A-27F are example screens of an interview system of the invention in accordance with one embodiment of the invention;

FIG. 28 is a flow chart of the steps used in the invention in accordance with one embodiment of the invention;

FIG. 29 is an example of an interform system of the invention in accordance with one embodiment of the invention;

FIG. 30 is an example of an interform system of the invention in accordance with one embodiment of the invention;

FIG. 31 is a example of the commands used to operate the interform system in accordance with one embodiment of the invention;

FIG. 32 is a flow chart of the steps used in the invention in accordance with one embodiment of the invention;

FIGs. 33A-33B are charts that illustrate the step used by the interform system in accordance with one embodiment of the invention;

FIGs. 34A-34B are charts that illustrate the step used by the interform system in accordance with one embodiment of the invention;

FIG. 35 is a chart that illustrates the step used by the interform system in accordance with one embodiment of the invention; and

FIG. 36 is a flow chart of the steps used in the invention in accordance with one embodiment of the invention.

Detailed Description of the Drawings

The method for providing bi-directional propagation among data within spreadsheets involves the steps of: a) determining if a circular reference has been defined; b) when a circular reference has been defined, determining if a user wants to be able to directly modify a calculated value; and c) when the user wants to be able to directly modify the calculated value, displaying a plurality of rule choices. The user selects (defines) the rules that are used to resolve the circular reference. In this way the user is aware of the rules and can make intelligent selections. In addition, this method can be used with even the most complex equations.

FIG. 1 is a front view of a computer 50 capable of operating the invention in accordance with one embodiment of the invention. The computer system 50 includes a monitor 52 for viewing input and output information. A mouse 54 (track ball, track pad, etc.) and keyboard 56 are used to provide input and view results. The computer 50 includes a processor 58 connected to a memory 60. In one embodiment the memory may contain a spreadsheet application 62. The spreadsheet application 62 is used by the analysis package to perform mathematical operations and for storing data.

FIG. 2 is a block diagram of the invention in accordance with one embodiment of the invention. In one embodiment the analysis package is structured in a client server model. A client application 70 requests data updates 72 and transmits data change requests 74 to a server application 76. The server application 76 has access to a data store 78. The client 70 controls a display 80.

FIG. 3 is a block diagram of the invention in accordance with one embodiment of the invention. The analysis package 90 includes an input system 92 that receives a plurality of data. An administrative system 94 receives the plurality of data from the input system 92. A data storage and scenario system 96 receives the plurality data and forms a scenario case. An output system 98 connected to the administrative system displays the scenario case.

FIG. 4 is a block diagram of an input system 110 of the invention in accordance with one embodiment of the invention. The input system 110 includes a keyboard 112 and mouse (track ball, etc.) 114 that can provide data directly to the administrative system 116 or to an interview subsystem 118. The interview subsystem 118 queries the user for some of the most relevant data for a given scenario, such as a software development project. In one embodiment, the interview system would ask questions such as: the number of programmers working on a project? and what percentage of their time is devoted to the project? Files 120 can also be imported to provide input data.

FIG. 5 is a schematic diagram of a data storage and scenario system 140 of the invention in accordance with one embodiment of the invention. The data storage and scenario system 140 includes a directory sheet 142 that lists every sheet in the data storage system 140. In one embodiment the directory sheet also provides other information about the sheets. such as title and location. A rules sheet 144 includes a sheet containing all the rules necessary to change data (a datum, modify a datum). In one embodiment the data storage and scenario system 140 has a plurality of rules sheets. One of the plurality of rules sheets includes a plurlaity of rules that are

implemented if a user wants to change a calculated value. Other rules sheets deal with adding/deleting data or requirements. The data storage system 140 includes at least one data sheet 146. The data sheet 146 includes both raw input and calculated values. The plurality of data sheets 146 form the scenario case. In one embodiment, the plurality of data sheets 146 includes a financial analysis data sheet, a project analysis data sheet and a market analysis data sheet. The plurality of data sheets 146 include a plurality of links between the data sheets 146. In one embodiment, there is a separate output view for each of the plurality of data sheets. The plurality of data sheets 146 includes a lock sheet (locking mechanism, data lock sheet) for each standard data sheet. The lock sheet stores which data items have been locked by a user. In one embodiment the data storage and scenario system 140 includes a default data sheet 148. In one embodiment there is more than one set of default data sheets 148. A set of default data sheets 148 will corresponds to a specific business situation. For example one set of default data sheets corresponds to software development projects. Note that there may be a different interview subsystem for each set of default data sheets.

FIG. 6 is a flow chart of the steps used in operating the invention in accordance with one embodiment of the invention. The process starts, step 160, by entering an input at step 162. Next, it is determined if the input is a datum addition at step 164. When the input is a datum addition, the analysis package determines one of a plurality of data sheets that holds a datum of the datum addition at step 166. The datum of the datum addition is stored in the one of the plurlaity of data sheets at step 168. At step 170 a view associated

with the one of the plurality of data sheets is updated, which ends the process at step 172. In one embodiment, every view effected by the datum addition is updated.

In one embodiment, the input is a datum change. The analysis package finds a rules sheet associated with the datum change. The analysis package implements the plurality of rules required by the datum change to update at least one of the plurality of data sheets. A view associated with the at least one of the plurality of data sheets is updated. In one embodiment, every view effected by the datum change is updated. Note that the analysis package allows a user to change calculated values as well as raw data. In one embodiment, the analysis package determines if one of the plurality of rules requires modifying a locked datum. When a rule requires modifying a locked datum, the user is sent a message. The message can be an error message and explain that a particular datum is locked. In another embodiment, the plurality of rules include a first priority set of rules if no data is locked, and a second priority set of rules if one of the datums is locked. The sets of rules will cover every possible permuatation of locked and unlocked data.

In one embodiment, the input is a datum deletion. The analysis package then determines one of the plurality of data sheets that holds a datum of the datum deletion. The analysis package then deletes the datum of the datum deletion from the one of the plurality of data sheets. A view associated with the one of the plurality of data sheets is updated. In one embodiment, every view effected by the datum deletion is updated.

In one embodiment, a user selects a type of product development they wish to analyze, for example, a software product, a

semiconductor product or a telecommunication project. The analysis package then uses a plurality of default data sheets associated with the type of product development. In one embodiment an interviewing subsystem associated with type of product development is initiated. The interview subsystem receives a plurality of answers from a user.

In one embodiment, the input is a datum lock. The analysis package stores a rule that a datum associated with the datum lock is locked.

In another embodiment, the user manipulates a graphical image to make a change to a datum.

FIGs. 7-12 are example screens of an interview system of the invention in accordance with one embodiment of the invention. FIG. 7 shows the general view 180 of the interview subsystem. This view allows a user to enter basic project data such as the name of the project 182, the start date of the project 184, the end date of the project 186, the labor burden rate of the resources assigned to work on the project 188 and the overhead burden rate of the resources assigned to work on the project 190. Note the lock icon next to the data input sections. When the user clicks on the lock the data is locked and cannot be changed either by the user or by the system. The user can unlock the data by clicking on the lock again.

FIG. 8 shows the developers view 192 of the interview subsystem. This view allows the user to enter development information about the project. The view has several subviews. The user has selected the general information subview 194. Here the user enters the name of a developer 196, the monthly salary of the developer 198, and the annual salary increase of the developer 200.

FIG. 9 shows the maintainers view 202 and the general information subview 204. Here the user can enter the name of the maintainer 206, the maintainer's annual salary 208 and the annual salary increase 210.

FIG. 10 shows the market segments view 212 and the general information subview 214. Here the user can enter the name of the market segment 216, the market segment size 218, the rate at which the market segment will grow 220 and the target sales price 222.

FIG. 11 shows the sales view 224 of the interview system. Here the user can enter the cost per unit sold 226, the fixed cost per unit sold 228 and the market window 230.

FIG. 12 shows the requirements view 232 and the general information subview 234. Here the user can enter a requirement ID 236, a description of the requirement 238, the number of person hours required to complete the requirement 240, the development risk 242 and whether to enter the requirement into the baseline 244.

FIGs. 13-14 are example screens of an output system of the invention in accordance with one embodiment of the invention. FIG. 13 shows an output graph that plots the probability of completing the project versus the number of person months. A line 260 represents the planned number of person months. The user can edit the planned person months by dragging the line to a new position 262. FIG. 14 shows the graph after the change. Thus the invention allows a user to manipulate a graphic image to provide input to the analysis package.

FIG. 15 is an example screen 280 of an output system of the invention in accordance with one embodiment of the invention. The screen 280 shows four views: the requirements view 282; the development personnel view 284; the risk assessment view 286 and

the effort estimation view 288. These views can be displayed (removed) by clicking on the associated icons 288.

FIG. 16 is a flow chart of the steps used in resolving circular references in accordance with one embodiment of the invention. The process starts, step 300, by determining if a circular reference has been defined at step 302. When the circular reference has been defined, the analysis package determines if a user wants to be able to directly modify a calculated value at step 304. When the user wants to be able to directly modify the calculated value at step 306, the system displays a plurality of rule choices which ends the process at step 308. In one embodiment, the user selects one of the plurality of rule choices. The plurality of rules are incorporated into a rules sheet. The user may then select a second of the plurality of rules choices. The second of the plurality of rule choices are incorporated into the rules sheet. Note that the process of FIG. 16 can be part of the analysis package described in FIGs. 1-15 or the process can stand alone. In a stand alone mode it would commonly be used with a spreadsheet application, however other applications can also be used.

In one embodiment, a first priority flag is associated with the one of the plurality of rule choices and a second priority flag is associated with the second of the plurality of rule choices.

In one embodiment, the process displays a rule that locks a selected variable. In another embodiment, the process displays a rule that a selected variable varies directly with the calculated value. In another embodiment, the process displays a rule that a selected variable varies in a preselected way with the calculated value. In another embodiment, the process displays a rule that a selected variable is determined based on a control equation.

FIG. 17 is a flow chart of the steps used in resolving circular references in accordance with one embodiment of the invention. The process starts, step 320, by determining if a value is a calculated value at step 322. When the value is a calculated value, the process determines the variables used to calculate the calculated value at step 324. The system then displays rule choices to the user at step 326. These first rules choices do not have any locked data or range limitations. The process then receives the user's selection as input at step 328. Next, the process displays the lock rule choices at step 330. The lock rules choices define how the process adjusts the remaining unlock variables when one or more variables are locked. The system requests that the user define a rule for every possible combination locked variables. The system receives the user's input at step 332. The system then displays the range rule choices at step 334. The user can select not to limit the range of any of the variables. However, when a user does limit the range of a variable, the process asks for a rule when the variable hits a limit of its range. The process receives the user's input which ends the process at step 336. Note that the process of FIG. 17 is used in setting up the rules for resolving circular references. A second part of the invention is how the rules are implemented when a user changes the calculated value. The implementation part of the invention is explained in more detail with respect to FIG. 19.

FIG. 18 is a list of rules used in resolving a circular reference in accordance with one embodiment of the invention. Note that the rules given are examples and other rules can also be defined. The calculated value is "F" and the variables are "A", "B", "C", "D", and "E". In this example the user is first asked to apportion the change in F

(ΔF) among the change in A (ΔA), the change in product BC (ΔBC), and the change in exponential D^E (ΔD^E). The user then defines a percentage (a, b, d) for each of the "sum variables". Note that if none of the "sum variables" are locked then a, b, d must add up to be one. The user then proceeds to define these percentages if any of the "sum variables" are locked. Next the user is required to define how to apportion the change in B (ΔB) compared to the change in C (ΔC). Commonly with an exponential, one or the other of the variables is locked. However, other rules are possible. Next, the user is asked if he wants to limit the range of any of the variables. For instance, a user may never want a variable to have a value of less than zero. If a situation arises where the variable would normally be negative, the user is asked how the other variables are to be adjusted.

FIG. 19 is a flow chart of the steps used in implementing the rules to resolve a circular reference in accordance with one embodiment of the invention. The process starts, step 340, by determining if a calculated value has been changed at step 342. When a calculated value has been changed, the sum rules are applied at step 344. Note that the appropriate sum rule has to be selected based on which variables are locked. Next it is determined if the sum rule violates a range rule at step 346. When the sum rule does violate the range rules, an out of range rule is applied at step 348. When no range rule is violated, the product rules are applied at step 350. The process then determines if the product rule violates a range rule at step 352. When the product rule violates a range rule, the process applies the out of range rules at step 356. The process ends at step 356. Note that an exponential rule could also be applied with checks for violations of range rules.

The following descriptions will focus on the preferred embodiments of the present invention, which are embodied in spreadsheet applications operative in the Microsoft Windows environment. The present invention, however, is not limited to any particular application or to any particular environment. Instead, those skilled in the art will find that the systems and methods of the present invention may be advantageously applied to a variety of systems software over a variety of different platforms. Therefore, the descriptions of the exemplary embodiments that follow are for purposes of illustration and not limitation.

The present invention is a system and method for creating a relationship between a plurality of data and a graphic representation of that data. When data is stored in a spreadsheet, the invention displays that data in a graphical view. This invention allows decision-makers to quickly visualize the data in a graphical format and use that data to make informed decisions.

The invention is embodied in a tool that can be used by decision-makers to see multiple, graphical views of their data.

The invention has a number of aspects of particular interest to the user: the opening and closing of graphical views, the entering of data, the presentation of graphics, the modifiability of the data. the viewing of data, and the propagation of changes.

System Hardware

FIG. 20 is a block diagram of an exemplary computer system 400 suitable for practicing the invention. The computer system 400 comprises a computer 402 having a processor 404, memory

(typically including both Random Access Memory (RAM) and at least one disk drive). A spreadsheet 406 is stored in the memory 408. The computer system 400 also includes a display 410, and a pointing device 412, which may be a mouse (other pointing devices, such as a trackball, stylus, light pen, or touch screen may also be used) and a keyboard 414.

FIG. 21 is a block diagram of an exemplary system 420 in accordance with the invention. The system 420 consists of a server 422 and a client 424 that exchange data. The server 422 has a storage space for data 426 in its memory. The user of the invention interacts with the invention through using the input devices (pointing device 412 and keyboard 414) and viewing the display 410 that is connected to the client 424.

The invention is a system and method for managing a plurality of data values that represent graphical views. The user of the invention shall interact with the invention through a process.

Interface

The user of the invention shall interact with the invention through the use of a mouse 412, a keyboard 414, or other input device. The preferred interface with the invention is shown in FIGs. 23A-23B and FIG. 24.

Interacting with Graphical Views

The user of the invention must be able to open and close the plurality of graphical views. To this end, the user may use the pointing device 412 or keyboard 414. Once the instance of the invention is displayed 500 on his display device 410, the user may use his mouse 412 to point to the view icons 506 in the view

bar 502. When the user clicks on a view icon 506 with his pointing device, the corresponding graphical view opens 508, and the view icon now has a dark background 504 to serve as a visual cue to the user that the view is currently being displayed.

The user of the invention may also use the keyboard 412 to open and close graphical views. To do this, he chooses Window from the main menu (by holding down the Alt key while pressing the W key) and then selects the graphical view to open (by using the arrow key to highlight it in the list and pressing the Enter key).

The user may have any number of views open at a time. If the user of the invention has pressed the auto-tile button 510, then when the user opens and closes graphical views, the views will automatically tile in the working space. In this context, "automatically tile" means that each view will take up as much space as all other open views and that the view will be able to be seen completely. If the user of the invention has not pressed the auto-tile button 510, then when the user opens and closes graphical views, the views will simply open in the working space. and the user will be able to move them and resize them to suit his viewing preferences.

Entering the Data

The specific data that the user must enter into the invention are the factors needed to model the business decision that they are trying to make. The user must have sufficient general understanding of the factors, and their interdependencies, which

make up the parts of the business decision. The data items that are editable by the user are called "primitive" data items.

There are two ways to enter primitive data 440 directly into the invention: by manipulating a graph, or by entering data directly into a data entry screen. As shown in FIGs. 23A and 23B, the user of the invention may manipulate a graphical item directly by using a pointing device to point to the item 462 and dragging that graphical item to a new position 464. The value of the "Planned Person Months" item was 24 months before the users edited it (see FIG. 23A, 466), and 36 months after the user edited it (see FIG. 23B, 470). As shown in FIG. 24, the user of the invention may enter data directly into editable areas 482, 484, 486, 488, 490 of the data entry views 480 by typing or using a pointing device. There are several data entry views (known in this instance of the invention as the Interview 480) into which the user may enter primitive data values. The data entry view shown in FIG. 24 is merely one instance of such a view.

Presenting the Graphical Views

The invention must determine the layout of the graphics on the screen. The invention displays each graph (which is a view of the underlying data) in a standard Microsoft® Window's window. For the layout (tiling) of the windows, it can either (1) automatically enforce the tiling of the windows so that the user can always view all of the data that is displayed, or (2) follow the conventional Windows default for tiling the windows. The preferred embodiment is the second.

Modifying the Data

After the user of the invention has entered the primitive data 440 and opened the graphical views 442, he may want to make changes to the data 444. The invention allows the user to modify the underlying data 444 by (1) allowing the user to modify the underlying spreadsheet directly, or (2) allowing the user to try to edit selected parts of any graph, and allowing the spreadsheet to use it's "seek" function to search for constants that it could change that would cause the graph edit to occur, or (3) allowing the user to edit any part of the graph and automatically propagate that change to the underlying data using InterForm, or (4) allowing the user to edit underlying data by means of an Interview that shows the user all of the primitive data elements in the data model. The preferred embodiments are the third and fourth.

If the user were allowed to modify the data by method (1), modifying the underlying spreadsheets directly, there would be little gain over the already-popular spreadsheet programs. If the user were allowed to modify the data by method (2), editing graphs and allowing the spreadsheet to use it's "seek" function. there would be a considerable loss of control, as the user would not be allowed to determine which values should change based on changes to constant values.

This instance of the invention allows the user to modify the data by means of editing graphical objects directly, as well as by allowing the user to edit primitive data values in the Interview 480.

The user can interact with the invention by manipulating graphical objects 462 in a graphical view 460 by using a mouse

412 or other pointing device. The user can move the graphical object 462 shown in FIG. 23A by passing the mouse pointer over the object until the shape of the mouse pointer changes into an East-West arrow. The user can then edit the value of the item 462 by pressing the left mouse button and moving the mouse to the right 464 until the line is in the new desired position and releasing the left mouse button shown in FIG. 23B. This has the effect of changing the value of the underlying data from 24 person months 466 to 36 person months 470.

Step 444 can be carried out in iteration with step 446, until the user is satisfied that they have all of the information needed in order to make a sound decision 448.

Extending this method of interaction with the invention, the user may move any graphical object in any view, and instantly see the ramification of that change to all other affected data in all other views. This capability allows the user to perform illuminating what-if and scenario analysis on his data; saving snapshots of his data model at any time for future comparison.

Viewing the Data

Step 446 consists of observing multiple views of the data to analyze it and make a business decision. Some people are better at understanding data when they see it in a textual format, while others comprehend graphs more easily. During this step, the user can view the data in a textual format (FIG. 24), or in a graphical format (FIGs. 23A & 23B). There are many graphical views available to the user of the embodiment of the invention.

The user of the invention can open and close views by clicking with the pointing device 412 on the icons 504 on the view bar 502. The view bar 502 lists all of the views available to the user. The views are displayed to the user in the window 416 on the display 410. The user examines the views 508 to aid in his business decision.

In one embodiment, the invention is tailored to product development, specifically software development. The user must have sufficient general understanding of the factors, and their interdependencies, which make up the parts of the decision of which features to include in which release of a software product.

The user of the invention shall interact with the invention through a process. The preferred process is shown in FIG 26. In this process, the first step 520 for the user is to enter their basic data. The basic data items that are editable by the user are called "primitive" data items. Once the primitives have been entered, the user then observes the multiple views 522 as they display different perspectives of the data. At step 520, the preferred method for the user to enter the primitive data into the invention is to use the Interview (FIGs. 27A—27F).

The user can enter general project information on page one of the Interview FIG 27A. After the user clicks on the General tab of the Interview 540, he can enter basic project data such as the name of the project 542, the start date of the project 544, the end date of the project 546, the labor burden rate of the resources assigned to work on the project 548 and the overhead burden rate of the resources assigned to work on the project 550.

The user can enter development information about the project on another page of the Interview FIG 27B. After the user clicks on the Developers tab 560, and the General Information sub-tab of the Interview 562, he can enter data such as the name of the developer 564, the monthly salary of the developer 566, and the annual salary increase of the developer 568.

The user can enter maintenance information about the project on another page of the Interview FIG 27C. After the user clicks on the Maintainers tab 580, and the General Information sub-tab of the Interview 582, he can enter data such as the name of the maintainer 584, the monthly salary of the maintainer 586, and the annual salary increase of the maintainer 588.

The user can enter information about the market of the software product on another page of the Interview FIG 27D. After the user clicks on the Maintainers tab 600, and the General Information sub-tab of the Interview 602, he can enter data such as the name of the market segment 604, the potential market size 606, and growth rate of the market 608, and the price at which he plans to sell the product in this market segment 610.

The user can enter information about the expected sales of the software product on another page of the Interview FIG 27E. After the user clicks on the Sales tab 620, he can enter data such as the percentage cost per unit sold 622, fixed cost per unit sold 624, and the market window 626.

The user can enter information about the requirements for the software product on another page of the Interview FIG 27F. After the user clicks on the Requirements tab 630, and the General

Information sub-tab of the Interview 632, he can enter data such as an identification number for each requirement or feature 634, the text of the requirement 636, the effort estimation in hours for the requirement 638, the development risk for the requirement 640, and whether the requirement should be included in the current release of the software product or not 642.

Viewing the Data

Once the user has entered all of the primitive data 520, then he can observe the implications of that data 522. By viewing the information presented in these views, the user must now decide if this information is adequate for him to make a business decision or not 526. If the views, displaying the current data, do not provide the user with enough information to make a decision, or he does not like the values and implications he sees, then he can make modifications and add details to the basic data 524. This can be an iterative process 522, 526, 524, until the user is satisfied that the information he sees on the display 410 is enough information to help him make a sound business decision regarding the features to be included in a release of a software product 528.

Modifying the Data

Once the basic data has been entered into the system 520, the implications have been observed 522, and he decides that he would like to make changes to the basic data 526, he proceeds with step 524. During step 524, he is modifying data; that is. he is making changes and adding details to the data already entered in step 520.

The invention allows the user to modify the underlying data by (1) allowing the user to modify the underlying spreadsheet directly, or (2) allowing the user to try to edit selected parts of any graph, and allowing the spreadsheet to use it's "seek" function to search for constants that it could change that would cause the graph edit to occur, or (3) allowing the user to edit any part of the graph and automatically propagate that change to the underlying data using InterForm, or (4) allowing the user to edit underlying data by means of an Interview that shows the user all of the primitive data elements in the data model. The preferred embodiments are the third and fourth.

The user can interact with the invention to modify the data by manipulating graphical objects in a graphical view by using a mouse or other pointing device.

The user may move any graphical object in any view, and instantly see the ramification of that change to all other affected data in any other view. This capability allows the user to perform illuminating what-if and scenario analysis on his data; saving snapshots of his data model at any time for future comparison.

As shown in FIGs. 23A and 23B, the user of the invention may manipulate a graphical item directly by using a pointing device to point to the item 462 and dragging that graphical item to a new position 464. The value of the "Planned Person Months" primitive data item was 24 months before the users edited it (see FIG. 23A. 466), and 36 months after the user edited it (see FIG. 23B, 470).

Step 522 can be carried out in iteration with step 524, until the user is satisfied that he has all of the information needed in order to make a sound business decision 526.

Performing the Analyses

Step 522 consists of observing multiple views of the data to analyze it and make a business decision. During this step, the user can view the data in a textual format (FIGs. 27A – 27F), or in a graphical format. The user can look at the data from any of 17 viewpoints, including views that display: Features, Effort Estimation, Release Schedule, Risk Assessment, Development Personnel, Maintenance Personnel, Market Segments, Units Sold by Quarter, Quarterly Revenue, Distribution of Revenue, Cumulative Units Sold, Cumulative Revenue, Distribution of Cumulative Revenue, Unit Cost/Price/Margin, Labor, Sales vs. Service, Break Even Analysis.

In an instance of the invention, the "Feature View" of the user's data shows all of the information for each proposed feature and whether or not that feature is to be included in the current release. The "Effort Estimation View" of the user's data shows the probabilistic distribution of the cost to develop the features that have been included in the current release. The "Release Schedule View" of the user's data shows probabilistic distribution of the time it will take to develop the features which have been included in the current release. The "Risk Assessment View" of the user's data shows the relative schedule risk, technical risk, and development risk. The "Development Personnel View" of the user's data shows a list of all of the developers, along with their

start and end dates and the percent of their time that they will spend working on this software project. The "Maintenance Personnel View" of the user's data shows a list of all of the maintainers, along with their start and end dates and the percent of their time that they will spend working on this software project. The "Market Segments View" of the user's data shows detailed information for each of the market segments for this software product. The "Units Sold by Quarter View" of the user's data shows the projected units sold per quarter for this software product in all market segments. The "Quarterly Revenue View" of the user's data shows the projected revenue from sales and service per quarter for this software product in all market segments. The "Distribution of Revenue View" of the user's data shows projected distribution of revenue from sales and service per quarter for this software product in all market segments. The "Cumulative Units Sold View" of the user's data shows the projected cumulative units sold per quarter for this software product in all market segments. The "Cumulative Revenue View" of the user's data shows the projected cumulative revenue from sales and service per quarter for this software product in all market segments. The "Distribution of Cumulative Revenue View" of the user's data shows projected cumulative distribution of revenue from sales and service per quarter for this software product in all market segments. The "Unit Cost/Price/Margin View" of the user's data shows an analysis of product price, cost, and profit. The "Labor View" of the user's data shows details about the software projects labor costs. The "Sales vs. Service View" of the user's data shows relative sales and service income

by quarter. The "Break Even Analysis View" of the user's data shows when the financial break-even point will be realized for this software product.

Once the instance of the invention has been displayed to the user 500, the user of the invention can open and close views by clicking with the pointing device 412 on the icons 506 on the view bar 502. When the user clicks on a view icon 506 with his pointing device, the corresponding graphical view opens 508, and the view icon now has a dark background 504 to serve as a visual cue to the user that the view is currently being displayed. The view bar 502 lists all of the views available to the user; both open and closed. At the user's request, the views are displayed to the user in the window 416 on the display 410. The user examines the views 508 to aid in his business decision.

Propagating the Data Changes

Whether the user of the invention is modifying textual data or graphical data 650, the fundamental process is the same. After the user makes a modification to the data, the invention sends that suggested change to the underlying data model (InterForm) 652. InterForm makes the determination of whether that suggested change can be made within the constraints of the model. If the change can be satisfied within the model, InterForm makes the change 654 and propagates the ramifications of the change to all of the other affected data items. The last step is to update all of the views (both graphical and textual) with all of the newly updated values 656.

Internal Operation

The method of the present invention for multi-viewpoint decision support is summarized by the flowchart of FIG. 28. The steps are as follows. At step 650, the user makes a modification to a data item. This modification can be made to either a graphical data item or a textual data item (as in the Interview); both types of modifications are treated the same by the invention. This requested change is sent from the client 424 to the server 422.

The invention then determines the type of change that has been requested by the user of the invention. There are three types of changes that a user can request: addition of data, deletion of data, or modification of data. If the data change request is one of addition or deletion of data, the invention simply makes the structural change to the underlying spreadsheet 406 by adding or removing space for that data. If the data change request is one of modification of data, the invention sends this change request to the data store 408, where the InterForm 652 takes over.

The purpose of the InterForm portion of the invention is to determine if the requested change can be satisfied, and if so, to make that change. If the users requested data change can not be satisfied (as determined by InterForm), the InterForm will notify the server, and the server will pass that notification message along to the client and thus the user.

There are several steps that the InterForm must take to determine if the user's requested data modification could be satisfied or not. First of all, the InterForm must locate the rule for that requested change. Each data value that is editable by the user of the invention is associated with a rule that specifically

states the actions that need to be taken when this value changes. The InterForm must scan that list and make sure that all of the actions could be carried out (given the current state of the system). If that check is successful, then InterForm implements the change that was requested by the user 654, and carries out all of the actions associated with that change.

The InterForm portion of the invention then notifies the server 422, which notifies the client 424, that the change was successfully made. All of the views in the user's display 410 are then updated with the new data 656.

InterForm Steps

The purpose of the InterForm portion of the invention is to determine if the requested change can be satisfied, and if so, to make that change. If the users requested data change can not be satisfied (as determined by InterForm), the InterForm will notify the server, and the server will pass that notification message along to the client and thus the user.

There are several steps that the InterForm must take to determine if the users requested data modification could be satisfied or not. First of all, the InterForm must locate the rule for that requested change. Each data value that is editable by the user of the invention is associated with a rule that specifically states the actions that need to be taken when this value changes. The InterForm must scan that list and make sure that all of the actions could be carried out (given the current state of the system). If that check is successful, then InterForm implements

the change that was requested by the user 654, and carries out all of the actions associated with that change.

Some definitions are necessary before the InterForm algorithm is detailed. InterForm as a noun, refers to the InterForm function as written in Visual Basic in Microsoft Excel. A Rule Set 660, 690 is a set of all rules for a given User-Requested Change. A User-Requested Change is the value requested by the user of the invention. A Priority Rule Set is the set of all rules with a single priority within a Rule Set. A Rule Set can have one or more Priority Rule Sets. No more than one Priority Rule Set can be executed for each User-Requested Change. A Rule is a single row in a Rule Set, and equates to a single row of information in the "InterForm Rules" spreadsheet contained in Microsoft Excel. It may include all information and an Excel formula needed to execute a User-Requested Change either in whole or in part. Each row in a rule set defines a single address in another sheet that represents three things: the value in that cell, that cells lock value, and that cell's User Friendly Name code. User Friendly Name Code is an integer that points to a cell address that contains a descriptive definition of that cell's value.

The InterForm function receives a user-requested change through the parameters of its function 710. Six parameters 712, 714, 716, 718, 720, 722, are passed to InterForm from the server 422. These parameters contain all of the information needed to make the user-requested change.

In this first example (FIG. 29), assume that the user of the invention has made a request to change the value of the first

requirement's person hours from 1 to 1300. Upon receipt of this user-requested change 710, the InterForm portion of the current invention takes the following 26 steps.

Step (1) InterForm scans all rows of the InterForm_Rules sheet inspecting the Requested Sheet column (FIG. 29, column b) for a match with strRequestedSheet 712. If a match is found then nRequestedRow 714 is checked to see if it is in the range defined by the Requested Min Row and Requested Max Row (FIG. 29, columns c and d). If the match fails then InterForm continues scanning the sheet. If a match is found then nRequestedCol 716 is checked to see if it is in the range defined by the Requested Min Column and Requested Max Column (FIG. 29, columns e and f). If a match is found then InterForm has found a matching Rule Set for the Requested Change. If all rows of the spreadsheet are scanned and no match is found, then an error is raised.

Step (2) InterForm inserts nRequestedNewValue 718 in the Requested New Value / Priority column (622, column g) in the first row of the Rule Set. In this example the value is 1300.

Step (3) Runtime Row Offset and Runtime Column Offset are determined. Offsets are calculated based on the arguments passed in to InterForm (nRequestedRow 714 and nRequestedCol 716) and the Requested Min Row and Requested Min Column values in the first row of the Rule Set. Offsets are used to determine which of a large set of data is to be modified. In this example the first Requirement's Person Hours are being modified, therefore the row offset is 1. If the third Requirement's Person Hours were being modified then the Runtime Row Offset would be 3. The calculated

Row and Column offsets are entered into the Runtime Row Offset and Runtime Column Offset columns respectively (664, 666, 668, 670, 672, 674, 676, 678, 680, 682, 684, columns n and o).

Step (4) The Runtime Row Offset and Runtime Column Offset values are then entered into their respective columns for every rule in the current Rule Set.

Step (5) Each rule may or may not contain a formula in its Row Lock Status column (FIG. 29, column q). Once the Runtime Row Offset and Runtime Column Offsets are updated then the Row Lock Status for each row in the Rule Set is recalculated. A value of true indicates that the cell indicated by the address information (the Implicated Sheet column (FIG. 29, column i), Implicated Row column (FIG. 29, column j) and Implicated Column (FIG. 29, column k)) in that row is currently locked. A value of false indicates that it is currently unlocked.

Step (6) The first row of each Priority Rule Set (not including the very first row of the entire Rule Set) contains a formula in the Priority Lock Status column (664, 666, 668, 670, 672, 674, 676, 678, 680, 682, 684, column r). Once the Row Lock Status has been updated for all rows in a Rule Set then the Priority Lock Status for each Priority Rule Set is recalculated.

Step (7) InterForm examines the Priority Lock Status for the first Priority Rule Set. If that value is false then all rules in that Priority Rule Set are unlocked and therefore can be executed (if there are no data validation errors). If the Priority Lock Status is true, then at least one of the rules in the Priority Rule Set has a value that is locked, therefore none of the rules may be executed.

InterForm then skips to the next Priority Rule Set, if there is one, and repeats this step. In this example the first Priority Lock Status is false (662, column r).

Step (8) The first rule in a Priority Rule Set may or may not contain an Error check formula in the Error column (662, column u). At this point InterForm checks the value in that column. If the value is false or there is no value then there are no data validation or range errors and InterForm can continue. If there is an error code in the Error column then InterForm returns with a failed status to the calling program along with the error code. In this example there is no error code, therefore InterForm will continue processing.

Step (9) The fifth argument in the call to InterForm is bDoImplicated 720 which is a Boolean flag to indicate whether to do the Requested Change (true) or simply to see if it can be done, but not to actually do it (false). At this point bDoImplicated 720 is examined, and if, false InterForm returns with a success status (DidIt) to indicate that it is possible. If bDoImplicated is true then processing continues. In this example it is assumed that bDoImplicated is true.

Step (10) The first row of any Rule Set that has a Priority Rule Set that can be executed (i.e. has a Priority Lock Status = false) is a rule that is always executed. For example. if the first and second Priority Rule Sets are locked and the third Priority Rule Set is unlocked then the first row/rule is executed. InterForm will then jump to the third Priority Rule Set and continue executing the rules in that Priority Rule Set. So for this step InterForm saves

the current priority which corresponds to the first found unlocked Priority Rule Set.

Step (11) InterForm executes the first rule 662 in the Rule Set.

Step (12) InterForm jumps to the first rule of the current priority's Priority Rule Set 664. There is only one Priority Rule Set in this example.

Step (13) All rules after the very first executed rule are called Implicated Changes. At this point InterForm executes all Implicated Changes 664, 666, 668, 670, 672, 674, 676, 678, 680, 682, 684 for the current Priority Rule Set.

Step (14) For each Implicated Change InterForm determines who is responsible for calculating the change by inspecting the Who Makes Change column (FIG. 29, column h). If the value in this column is Excel then it is an automatic calculation handled by Excel. If the value is InterForm or InterForm_Recalc then the calculation is performed by the code in InterForm. The first rule is always an InterForm calculation. Implicated changes can be either InterForm's job or Excel's job.

Step (15) The values found in the Implicated Sheet column (FIG. 29, column i), Implicated Row column (FIG. 29, column j) and Implicated Column column (FIG. 29, column k) are used to create an Excel address. This address points to the cell in another sheet whose value will be modified either by Excel or by InterForm. If the values in the Implicated Row or Implicated column have 2 numbers separated by a colon then that value represents a range. Those values then become loop boundaries. In this example the

address for the first rule 662 is the Requirements sheet, row 16, column 3.

Step (16) The current value in the New Value column (662, column l) is copied to the Old Value column (662, column m). This value is also placed on an Undo stack.

Step (17) If the Who Makes Change (FIG. 29, column h) value is InterForm or InterForm_Recalc then there will be a formula in the New Value column. That formula is executed and the result is placed in the cell whose address was formulated by the Implicated Sheet, Implicated Row and Implicated Column values. In this example the formula evaluates to 1300. Therefore the value 1300 is entered in the Requirements sheet, row 16, column 3.

Step (18) If the Who Makes Change (FIG. 29, column h) value is InterForm_Recalc then InterForm recalculates the entire workbook. In this example the value is InterForm therefore the workbook is not recalculated.

Step (19) The Implicated Sheet (662, column i), Implicated Row (662, column j) and Implicated Column (662, column k) values are entered in an array that will be passed back to the calling program along with a success or failure status and the number of Implicated Changes recorded in the array. In this example 'Requirements','16','3' is entered into this array.

Step (20) Steps 15 – 19 are repeated for each Implicated Change 664, 666, 668, 670, 672, 674, 676, 678, 680, 682, 684 in the current Priority Rule Set.

Step (21) Once all Implicated Changes are executed a success status is entered in the array.

Step (22) The last row of each Priority Rule Set may or may not have a formula in the Cascade Status column (684, column s) that indicates whether or not the Requested Change could be fully absorbed by the current Priority Rule Set. If this value is false or there is no formula then the Requested Change was fully absorbed and processing is complete. If this value is true then the current Priority Rule Set is locked and InterForm is called again (with the same arguments) in an attempt to complete processing the Requested Change at the next priority. In this case the cell is blank, indicating no formula, therefore the Requested Change was fully absorbed.

Step (23) If the last row of the Priority Rule Set has a Cascade Status formula then the Cascade Value column is inspected (684, column t). If the value in this column is true then the entire workbook is recalculated before the Cascade Status is evaluated. If the value is false or the cell is empty then processing continues.

Step (24) Processing continues until (a) the Cascade Status for the current Priority Rule Set is false (or there is no Cascade Status) after all rules in the current Priority Rule Set have been executed thereby indicating that the Requested Change has been fully executed, or (b) there are no more Priority Rule Sets that can be executed, because there are no more or the rest are all locked. In this case InterForm returns a failed status to the calling program and the User-Requested Change is not executed.

Step (25) Upon returning from a recursed call to InterForm the previously locked Priority Rule Set is then unlocked.

Step (26) InterForm returns the array of Implicated Changes to the server 422 along with a success status and the number of Implicated Changes in the array. This array is returned as a Variant 724.

The server then notifies the client 424 that the change was successfully made. All of the views in the user's display 410 are then updated with the new data 656.

In this second example (FIG. 30), assume that the user of the invention has made a request to change the value of the planned person months from 24 to 28.3. Upon receipt of this user-requested change 710, the InterForm portion of the current invention takes the following 30 steps.

Step (1) InterForm scans all rows of the InterForm_Rules sheet inspecting the Requested Sheet column (FIG. 30, column b) for a match with strRequestedSheet 712. If a match is found then nRequestedRow 714 is checked to see if it is in the range defined by the Requested Min Row and Requested Max Row (FIG. 30, columns c and d). If the match fails then InterForm continues scanning the sheet. If a match is found then nRequestedCol 716 is checked to see if it is in the range defined by the Requested Min Column and Requested Max Column (FIG. 30, columns e and f). If a match is found then InterForm has found a matching Rule Set for the Requested Change. If all rows of the spreadsheet are scanned and no match is found, then an error is raised.

Step (2) InterForm inserts nRequestedNewValue 718 in the Requested New Value / Priority column (692, column g) in the first row of the Rule Set. In this example the value is 28.3.

Step (3) Runtime Row Offset and Runtime Column Offset are determined. Offsets are calculated based on the arguments passed in to InterForm (nRequestedRow 714 and nRequestedCol 716) and the Requested Min Row and Requested Min Column values in the first row of the Rule Set. Offsets are used to determine which of many types of data are to be modified. In this example the Runtime Row Offset and Runtime Column Offsets are 0.

Step (4) The Runtime Row Offset and Runtime Column Offset values are entered into their respective columns for every rule in the current Rule Set.

Step (5) Each rule may or may not contain a formula in its Row Lock Status column (FIG. 30, column q). Once the Runtime Row Offset and Runtime Column Offsets are updated then the Row Lock Status for each row in the Rule Set is recalculated. A value of true indicates that the cell indicated by the address information (the Implicated Sheet column (FIG. 30, column i), Implicated Row column (FIG. 30, column j) and Implicated Column (FIG. 30, column k)) in that row is currently locked. A value of false indicates that it is currently unlocked.

Step (6) The first row of each Priority Rule Set (not including the very first row of the entire Rule Set) contains a formula in the Priority Lock Status column (694, 696, 698, 700, column r). Once the Row Lock Status has been updated for all rows in a Rule Set

then the Priority Lock Status for each Priority Rule Set is recalculated.

Step (7) InterForm examines the Priority Lock Status for the first Priority Rule Set. If that value is false then all rules in that Priority Rule Set are unlocked and therefore can be executed (if there are no data validation errors). If the Priority Lock Status is true, then at least one of the rules in the Priority Rule Set has a value that is locked, therefore none of the rules may be executed. InterForm then skips to the next Priority Rule Set, if there is one, and repeats this step. In this example the first Priority Lock Status is false (694, column r).

Step (8) The first rule in a Priority Rule Set may or may not contain an Error check formula in the Error column (694, column u). At this point InterForm checks the value in that column. If the value is false or there is no value then there are no data validation or range errors and InterForm can continue. If there is an error code in the Error column then InterForm returns with a failed status to the calling program along with the error code. In this example there is no error code, therefore InterForm will continue processing.

Step (9) The fifth argument in the call to InterForm is bDoImplicated 720 which is a Boolean flag to indicate whether to do the Requested Change (true) or simply see if it can be done, but not to actually do it (false). At this point bDoImplicated 720 is examined and if false InterForm returns with a success status (DidIt) to indicate that it is possible. If bDoImplicated is true then

processing continues. In this example it is assumed that bDoImplicated is true.

Step (10) The first row of any Rule Set that has a Priority Rule Set that can be executed (i.e. has a Priority Lock Status = false) is a rule that is always executed. For example, if the first and second Priority Rule Sets are locked and the third Priority Rule Set is unlocked then the first row/rule is executed. InterForm will then jump to the third Priority Rule Set and continue executing the rules in that Priority Rule Set. So for this step InterForm saves the current priority which corresponds to the first found unlocked Priority Rule Set.

Step (11) InterForm executes the first rule 692 in the Rule Set.

Step (12) InterForm jumps to the first rule of the current priority's Priority Rule Set 694. There are two Priority Rule Sets in this example. The Priority Lock Status of the first Priority Rule Set is false therefore it will be executed.

Step (13) All rules after the very first executed rule are called Implicated Changes. At this point InterForm executes all Implicated Changes 694, 696, 698, 700 for the current Priority Rule Set.

Step (14) For each Implicated Change InterForm determines who is responsible for calculating the change by inspecting the Who Makes Change column (FIG. 30, column h). If the value in this column is Excel then it is an automatic calculation handled by Excel. If the value is InterForm or InterForm_Recalc then the calculation is performed by the code in InterForm. The first rule

is always an InterForm calculation. Implicated changes can be either interform's job or Excel's job.

Step (15) The values found in the Implicated Sheet column (FIG. 30, column i), Implicated Row column (FIG. 30, column j) and Implicated Column column (FIG. 30, column k) are used to create an Excel address. This address points to the cell in another sheet whose value will be modified either by Excel or by InterForm. If the values in the Implicated Row or Implicated column have 2 numbers separated by a colon then that value represents a range. Those values then become loop boundaries. In this example the address for the first rule 692 is the Wizard sheet, row 20, column 1.

Step (16) The current value in the New Value column (692, column l) is copied to the Old Value column (692, column m). This value is also placed on an Undo stack.

Step (17) If the Who Makes Change (FIG. 30, column h) value is InterForm or InterForm_Recalc then there will be a formula in the New Value column. That formula is executed and the result is placed in the cell whose address was formulated by the Implicated Sheet, Implicated Row and Implicated Column values. In this example the formula evaluates to 28.30. Therefore the value 28.30 is entered in the Wizard sheet, row 20, column 1.

Step (18) If the Who Makes Change (FIG. 30, column h) value is InterForm_Recalc then InterForm recalculates the entire workbook. In this example the value is InterForm therefore the workbook is not recalculated.

Step (19) The Implicated Sheet (692, column i), Implicated Row (692, column j) and Implicated Column (692, column k) values are entered in an array that will be passed back to the calling program along with a success or failure status and the number of Implicated Changes recorded in the array. In this example 'Wizard','20','1' is entered into this array.

Step (20) Steps 15 – 19 are repeated for each Implicated Change 694, 696, 698, 700 in the current Priority Rule Set.

Step (21) Once all Implicated Changes are executed a success status is entered in the array.

Step (22) If the entire Requested Change could not be absorbed in the current Priority Rule Set the value in the Cascade Status column (700, column s) would be true. The Cascade Value column (700, column t) is false therefore a recalculation is not necessary to evaluate the Cascade Status value. Assume this is the case for now.

Step (23) The current Priority Rule Set will now be locked. The value in the Priority Lock Status for the first Priority Rule Set will not be true (700, column r).

Step (24) InterForm is now recursively called with the same arguments. Certain values have changed as a result of the rules in the first Priority Rule Set.

Step (25) When InterForm inspects the Priority Lock Status of the first Priority Rule Set it will see that its value is true, therefore it will skip to the next Priority Rule Set 702, 704, 706, and inspect its Priority Lock Status which will be false.

Step (26) The first rule of the Rule Set 692 will be executed.

Step (27) InterForm will then jump to the priority 2 Priority Rule Set and continue execution.

Step (28) After executing all the rules in that Priority Rule Set it will inspect the Cascade Status value which will be empty indicating that the Requested Change was fully executed.

Step (29) InterForm will return an array with a success status to the calling program, which in this case is the first instance of InterForm.

Step (30) Since the recursed InterForm call returned with success, InterForm again returns the array with another success status to the server 210.

The server then notifies the client 424 that the change was successfully made. All of the views in the user's display 410 are then updated with the new data 656.

Inteform can be used separate from the analysis package described above. When interform is used separately the user must provide a spreadsheet (other application) to interform. To do this, the user imports their spreadsheet, complete with circular references, into the software program embodiment of the invention.

Invention Resolves Circular References in User's Spreadsheet

In the next step of the process, the invention resolves the circular references in the user's spreadsheet.

To this end, the invention must first examine the user's spreadsheet for circular references 730. This is a matter of

looking at each cell that contains a formula and keeping a list of all of the cells that are referenced by that cell. A simple tree is built to store this information. When the invention has finished viewing each cell, the entire tree can be scanned for sets of cells that cause a circle of references to be formed.

Now the invention must identify which cells are directly alterable by a user 732. If an alterable cell is part of a chain of circular references, then it must be resolved by the invention 770. The invention must remove the references from the cells in the circular reference that refer to the alterable cell 734 and remember the removed cell references 736. Whenever the user makes a change to a spreadsheet value, instead of the spreadsheet making that change directly, the invention must propagate the change based on the new, non-circular formulas 738.

A Specific Example

The example shown in FIG. 33A is a typical, common example of a user's spreadsheet that contains a circular reference. In this example, cell A 744 of the user's spreadsheet refers to cell B 746, which refers to cell C 748. The cyclical relationship is further illustrated in FIG. 33B. Cell A refers to cell B through relationship 750. Cell B refers to cell C through relationship 752. And cell C refers to cell A through relationship 754.

As a specific example of resolving circular references, refer to FIG. 35. The original circular formulas from the user's spreadsheet (column 742) are listed in FIG. 35. The new, non-circular formulas, determined by the invention, are listed in column 780.

Assume that the spreadsheet cell that is alterable by the user is cell A. Therefore the invention must break the circular chain between cells A and B 770. Links 752 and 754 must be replaced with new, non-circular formulas 770.

The original formula for cells B and C must be replaced because they both refer to cell A, which causes a circular reference.

The invention must determine the best way to replace the original, circular formulas. It can do this by (1) trying to resolve the circular reference by inference from the original formulas, (2) asking the user a series of questions to help determine the correct resolution of the circular reference, or (3) using a predefined set of rules and formulas. The preferred embodiment is the third, although all work.

One resolution of the original circular reference is shown by the formulas in 782 and 784.

User uses "improved" Spreadsheet

When the users manipulates the values in the "improved" (no circular references) spreadsheet, the invention takes over when the user suggests a change to a cell that was originally part of a circular reference chain.

As a specific instance of a user using the improved spreadsheet. see FIG. 36. Assume the user-alterable cell is cell A. Assume the initial value contained in cell A is 2, cell B is 4, and cell C is 6 790.

The user now suggests changing the value of cell A from 2 to 10 792. At this point, the invention takes over to change the values in cells B and C. There is an absolute value difference between the

old value of A (2) and the suggested new value of A (10) of 8. Therefore, the invention sets the value of x to 8. In accordance with the formula 782, the invention subtracts as much of x as possible without allowing B to go below 0. Therefore, the value in cell B gets set to 0. The value for x is now 4. In further accordance with the formula 782, the invention now adds the remainder of x to C. Therefore the value in cell C gets set to 10. And x is now equal to 0 794. The old circular reference problem was completely resolved by the invention.

Now that the invention has resolved the circular reference, the final step is for the spreadsheet program to take over the rest of the propagation of values. At this point the only cell that does not have a value is cell A. The spreadsheet uses its normal method of propagation to compute the formula in this cell. Therefore the new value for cell A is 10 796. This is the value that the user suggested initially.

Thus there has been described an analysis package and method that clearly shows the effect of decisions, that clearly shows trends and provides information about all aspect of the business.

The methods described herein can be implemented as computer-readable instructions stored on a computer-readable storage medium that when executed by a computer will perform the methods described herein.

While the invention has been described in conjunction with specific embodiments thereof, it is evident that many alterations, modifications, and variations will be apparent to those skilled in the art in light of the foregoing description. Accordingly, it is intended to

embrace all such alterations, modifications, and variations in the appended claims.

Claims

What is claimed is:

1. A method for providing bi-directional propagation among data within spreadsheets, comprising the steps of:

(a) determining if a circular reference has been defined;

(b) when a circular reference has been defined, determining if a user wants to be able to directly modify a calculated value; and

(c) when the user wants to be able to directly modify the calculated value, displaying a plurality of rule choices.

2. The method of claim 1, further including the steps of:

(d) selecting one of the plurality of rule choices;

(e) incorporating the one of the plurality of rule choices into a rules sheet.

3. The method of claim 2, further including the steps of:

(f) selecting a second of the plurality of rule choices;

(g) incorporating the second of the plurality of rule choices into the rules sheet.

4. The method of claim 3, wherein a first priority flag is associated with the one of the plurality of rule choices and a second priority flag is associated with the second of the plurality of rule choices.

5. The method of claim 1, wherein step (c) includes the step of:

(c1) displaying a rule that locks a selected variable.

6. The method of claim 1, wherein step (c) includes the step of:

(c1) displaying a rule that a selected variable varies directly with the calculated value.

7. The method of claim 1, wherein step (c) includes the step of:

(c1) displaying a rule that a selected variable varies by a preselected way with the calculated value.

8. The method of claim 1, wherein step (c) includes the step of:

(c1) displaying a rule that a selected variable is determined based on a control equation.

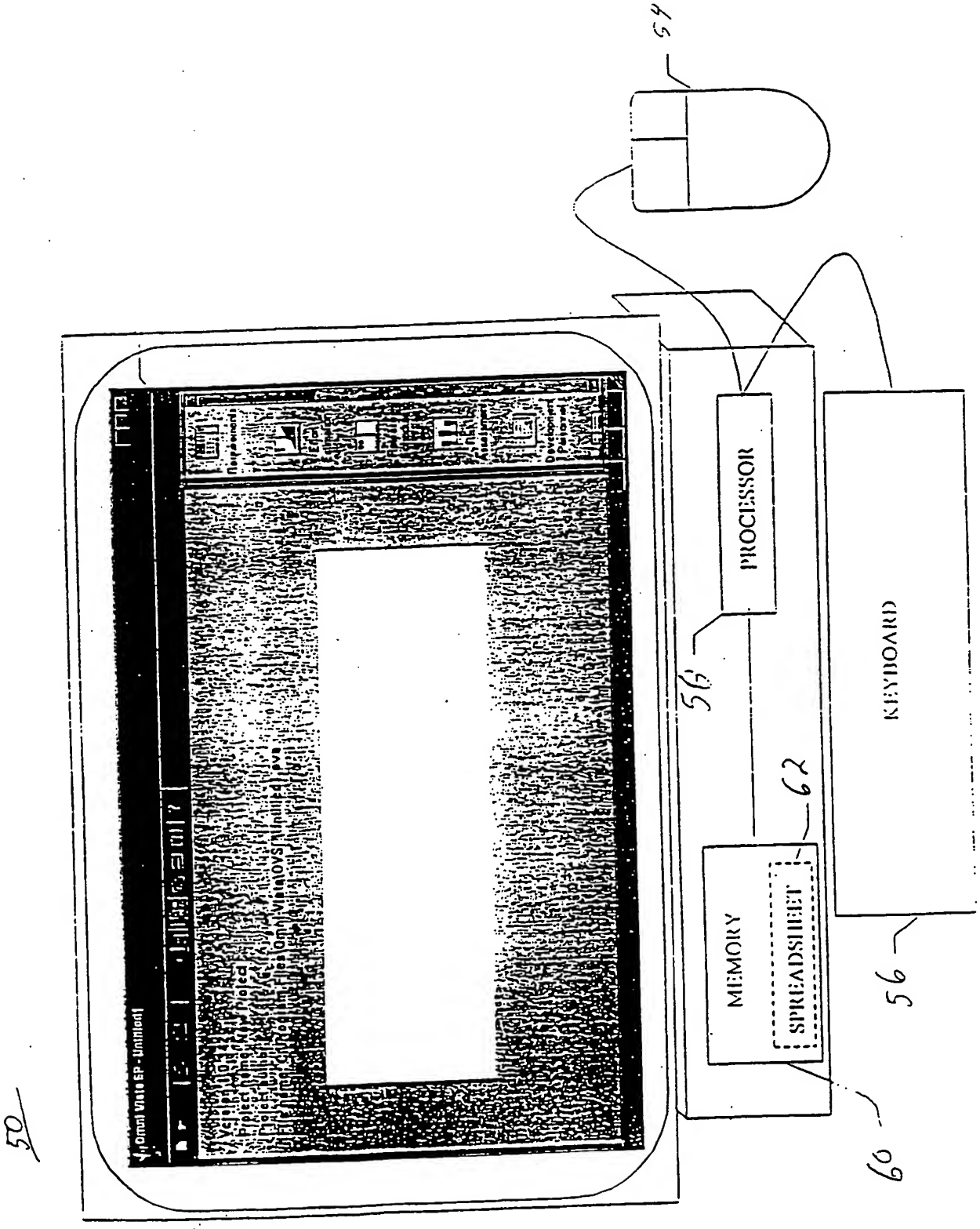


FIG. 1

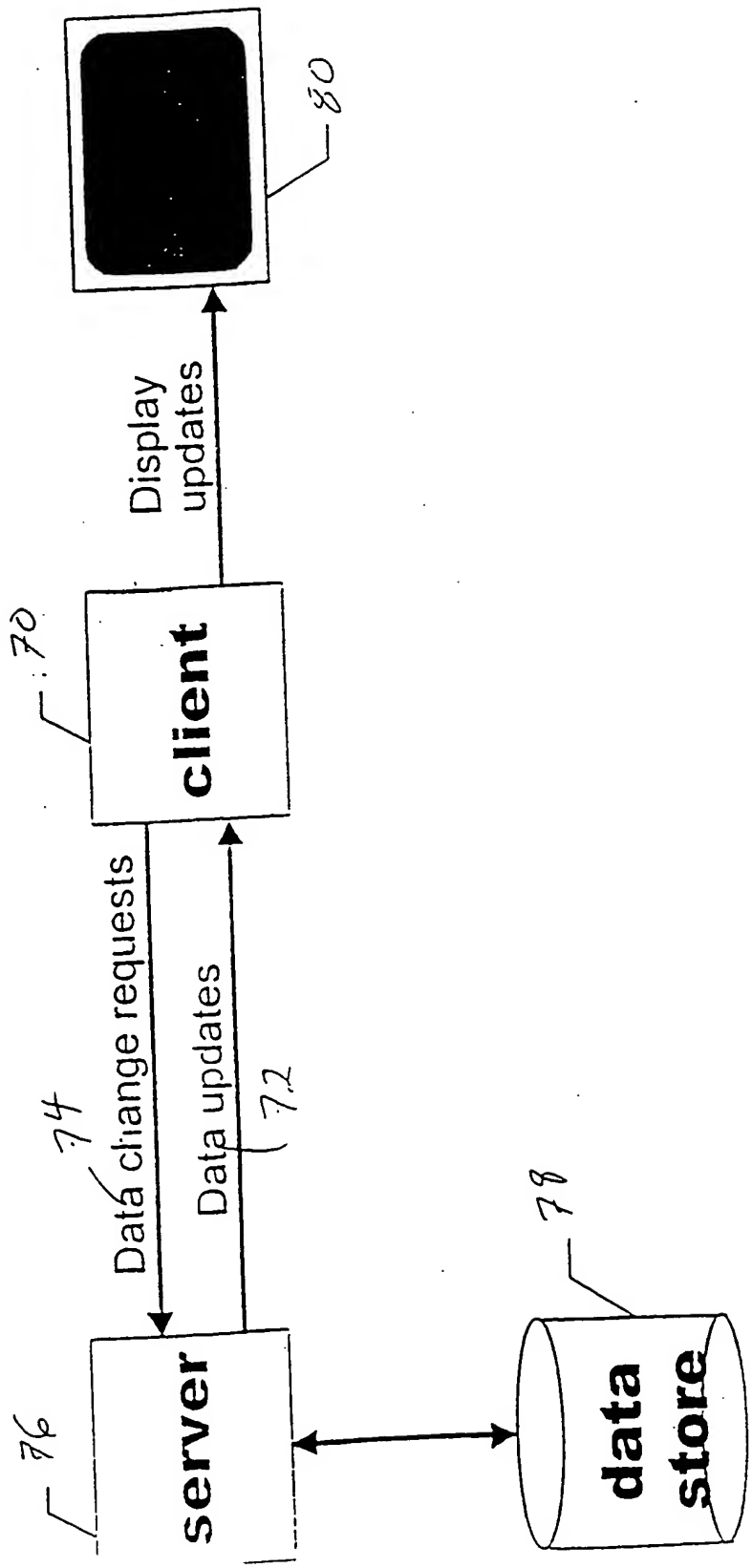

80
Display updates
70
client
Data change requests
74
Data updates
72
78
server
76
data store

FIG. 2

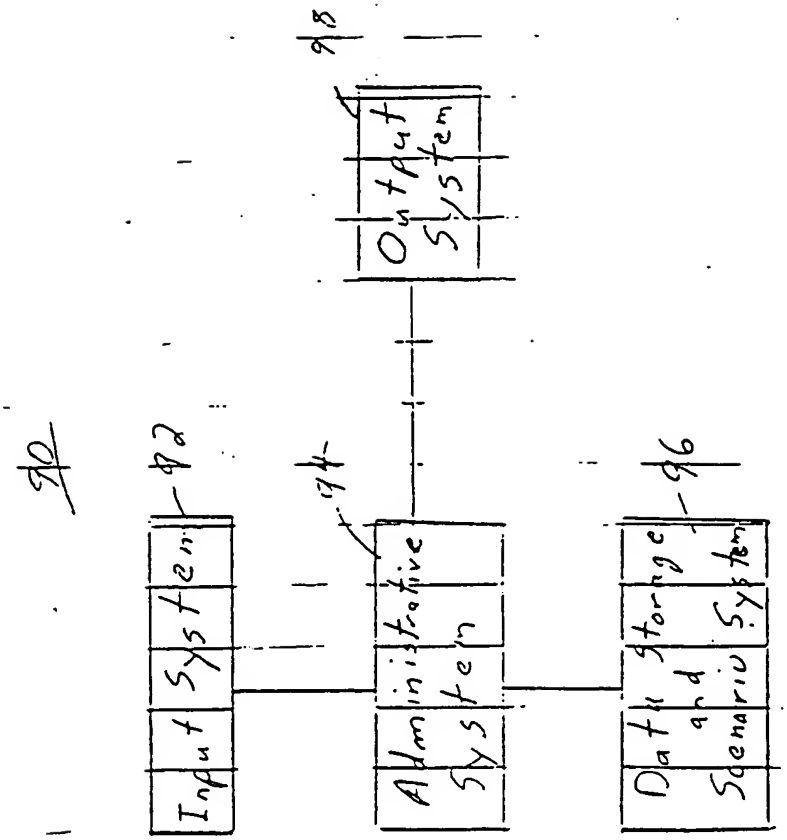

90
92
Input System
94
Administrative System
96
Data Storage and Scenario System
98
Output System

FIG. 3

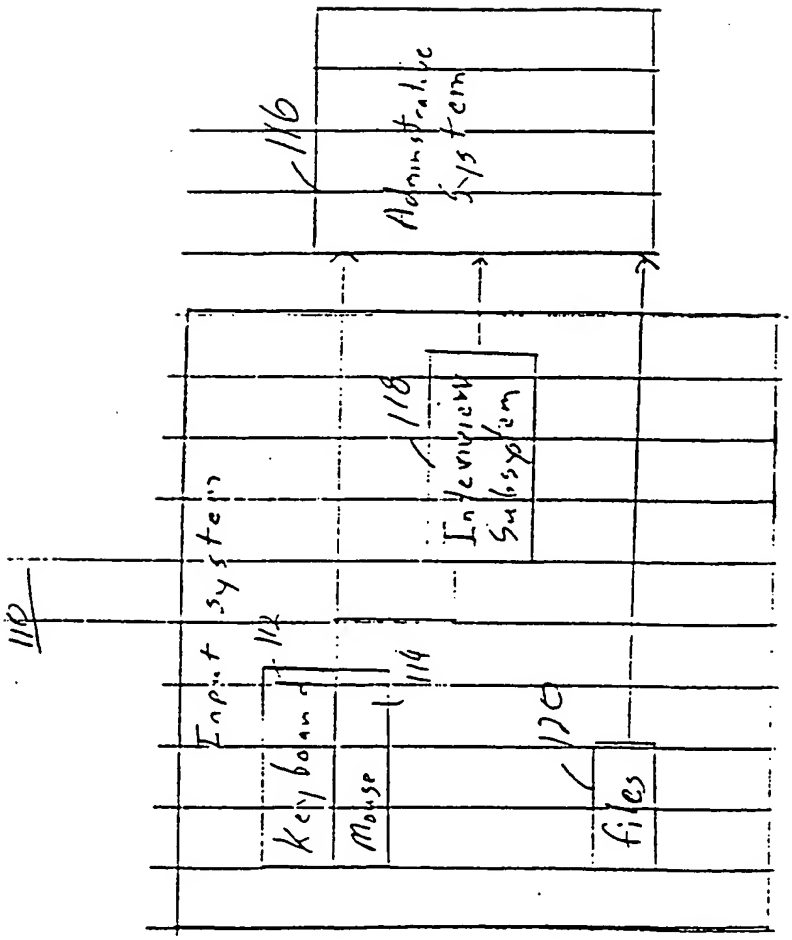

116
Administrative system
118
Interview Subsystem
112
Input systems
Keyboard
Mouse
114
120
Files

FIG. 4

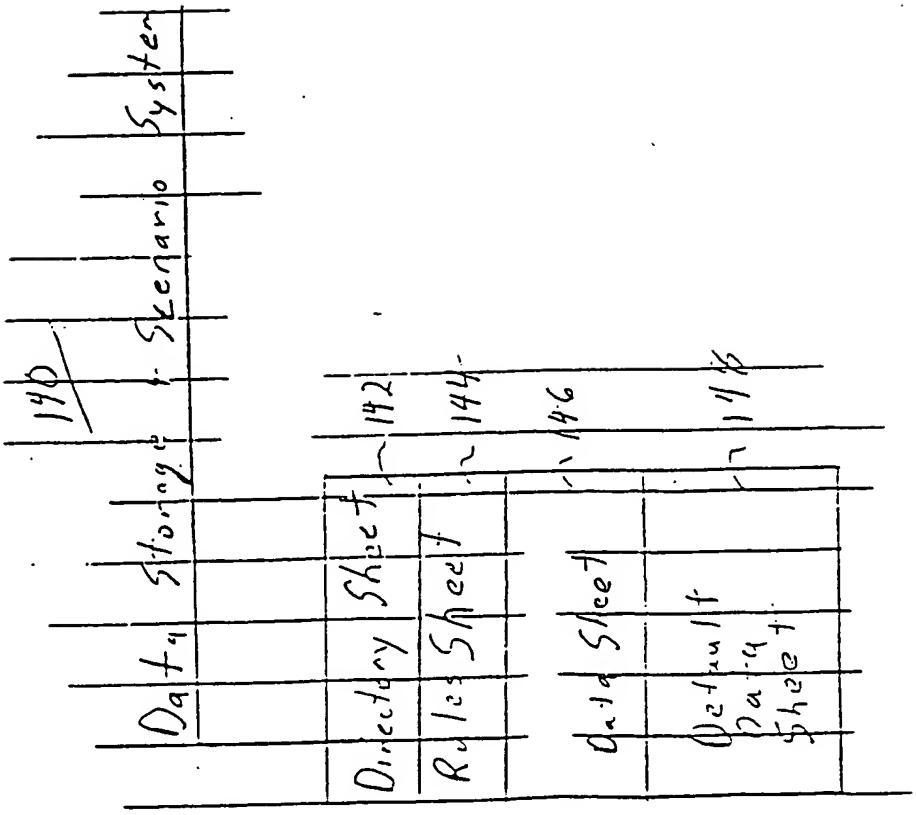

140
Scenario
System
Strategy
Data
142
Directory Sheet
144
Rules Sheet
146
Data Sheet
148
Default Data Sheet

FIG 5

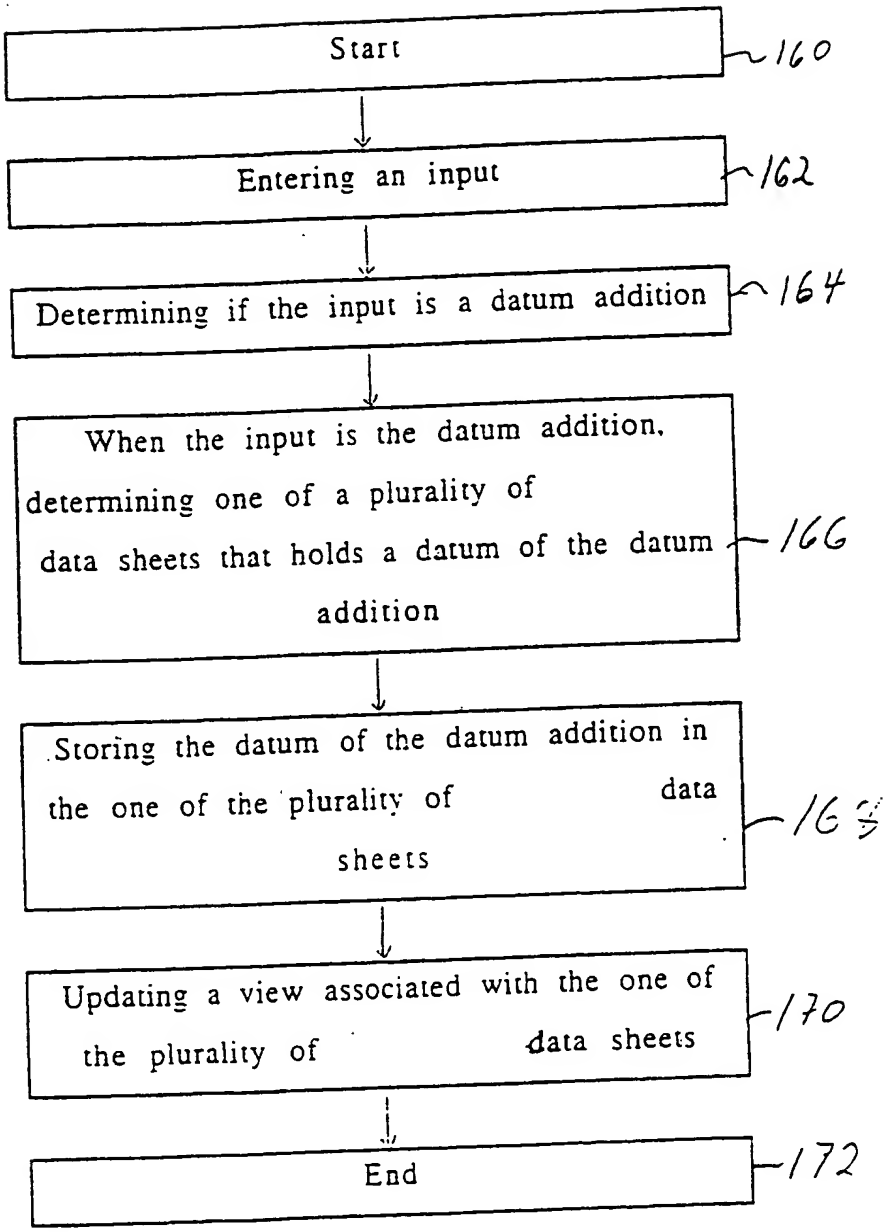

Start
160
Entering an input
162
Determining if the input is a datum addition
164
When the input is the datum addition, determining one of a plurality of data sheets that holds a datum of the datum addition
166
Storing the datum of the datum addition in the one of the plurality of data sheets
168
Updating a view associated with the one of the plurality of data sheets
170
End
172

FIG. 6

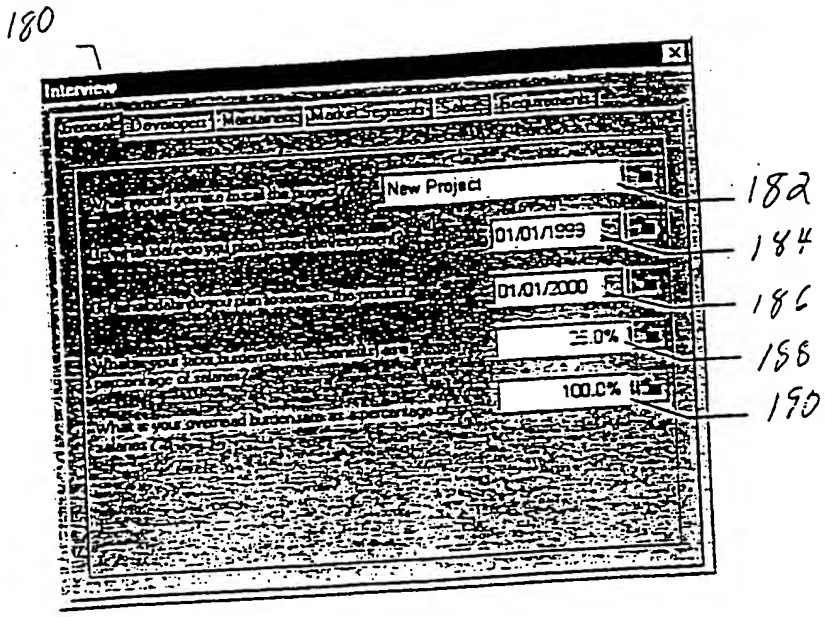

180
Interview
New Project
01/01/1999
01/01/2000
100.0%
182
184
186
188
190

FIG. 7

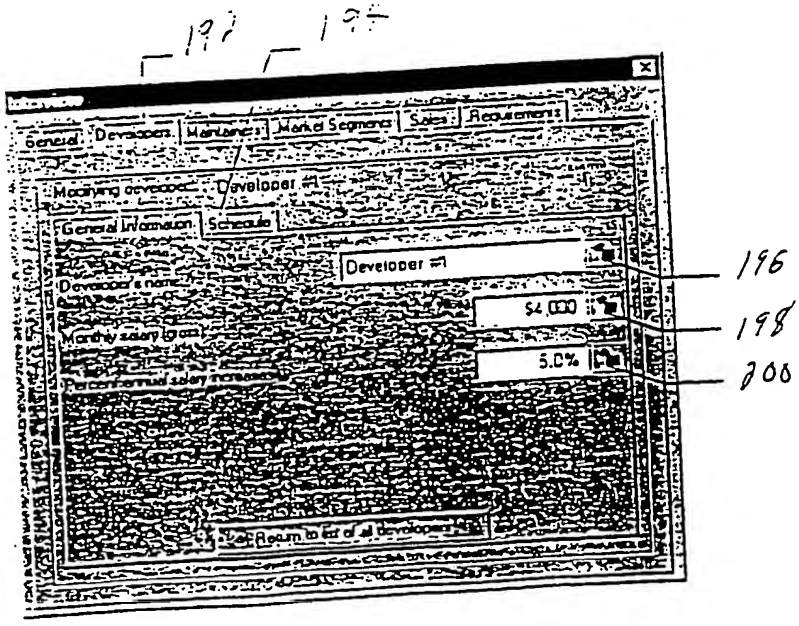

192
194
General
Developers
Maintenance
Market Segments
Sales
Requirements
Developer #1
General Information
Schedule
Developer #1
$4,000
5.0%
196
198
200

FIG. 9

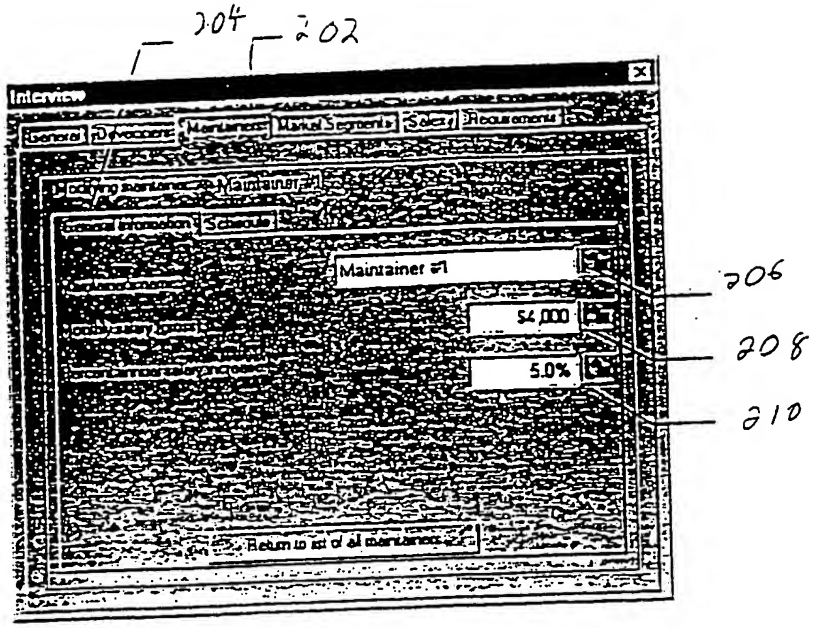

204
202
Interview
General
Maintainers
Market Segments
Sales
Requirements
Maintainer #1
General Information
Schedule
Maintainer #1
$4,000
5.0%
206
208
210

FIG. 9

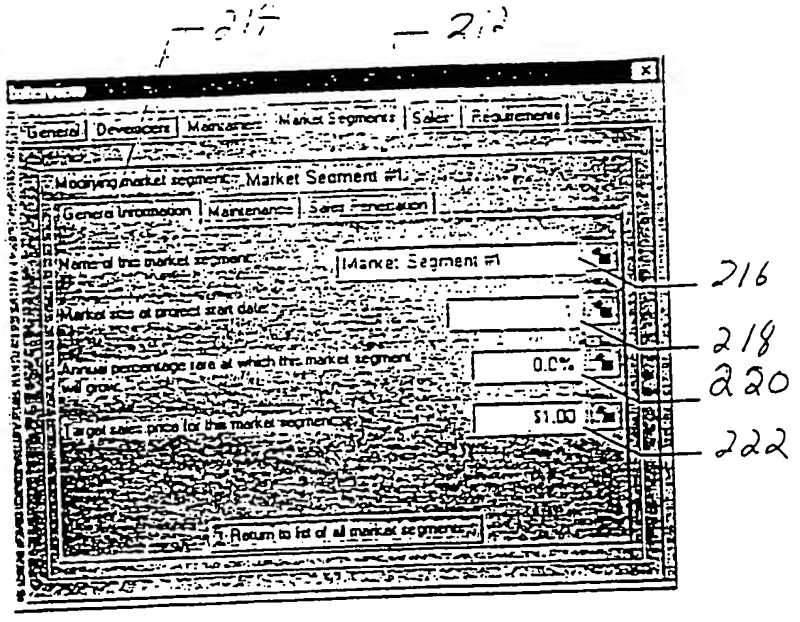

214
212
General
Developers
Maintainers
Market Segments
Sales
Requirements
Modifying market segment: Market Segment #1
General Information
Maintenance
Sales Projection
Name of this market segment:
Market Segment #1
Market size at project start date:
Annual percentage rate at which this market segment will grow:
0.0%
Target sales price for this market segment:
$1.00
Return to list of all market segments
216
218
220
222

FIG. 10

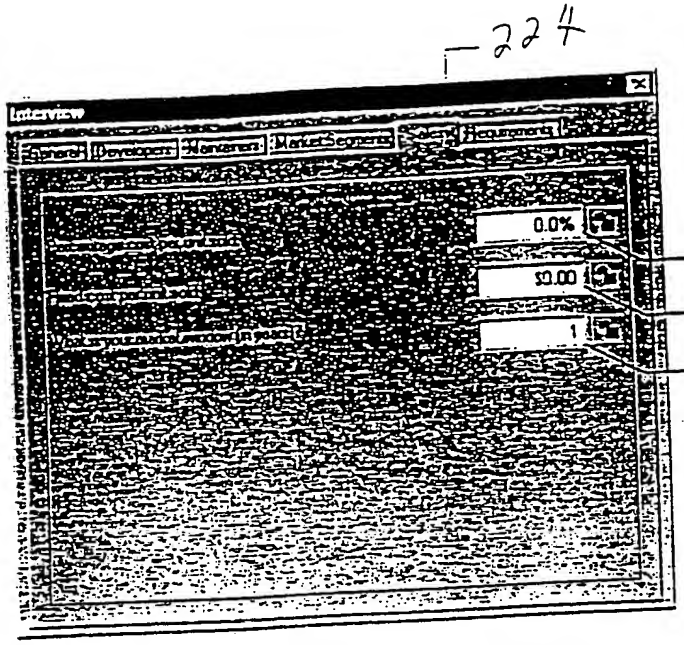

224
Interview
General
Developers
Managers
Market Segments
Requirements
0.0%
$0.00
1
226
228
230

FIG. 11

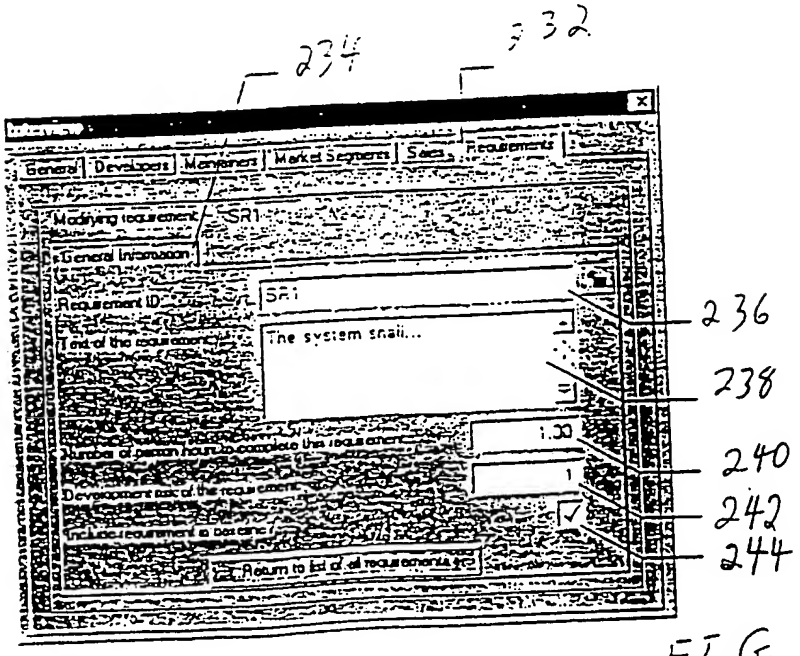

234
232
General
Developers
Managers
Market Segments
Requirements
Modifying requirement SR1
General Information
Requirement ID
SR1
Text of the requirement
The system shall...
1.00
1
Return to list of all requirements
236
238
240
242
244

FIG. 12

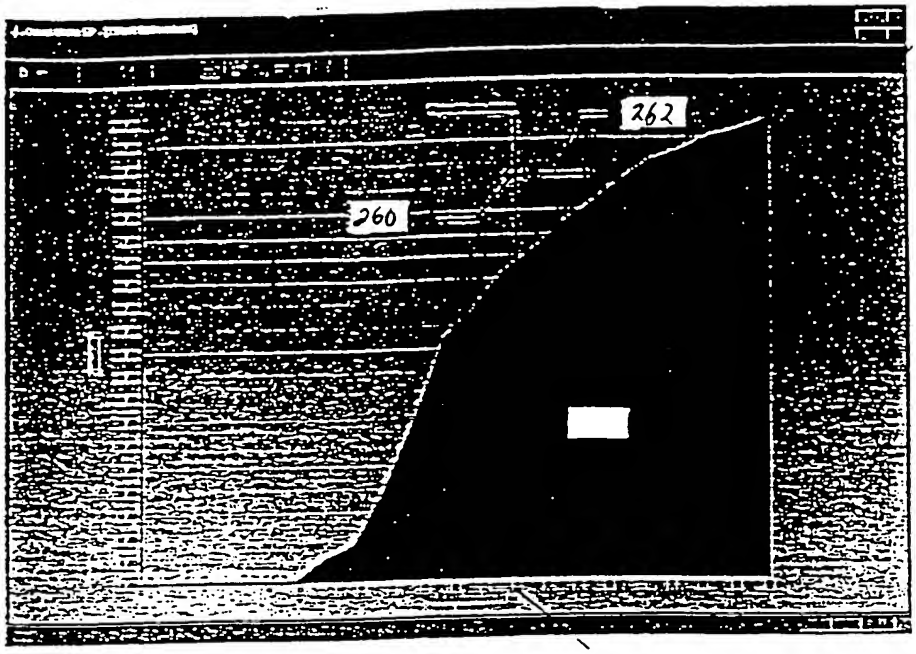


FIG. 13

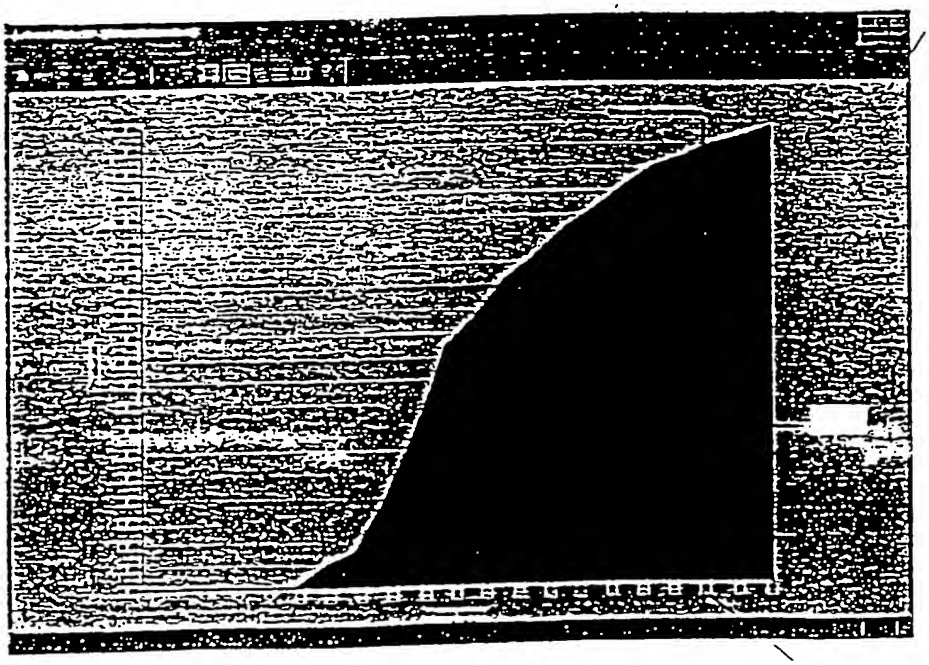


FIG. 14

280

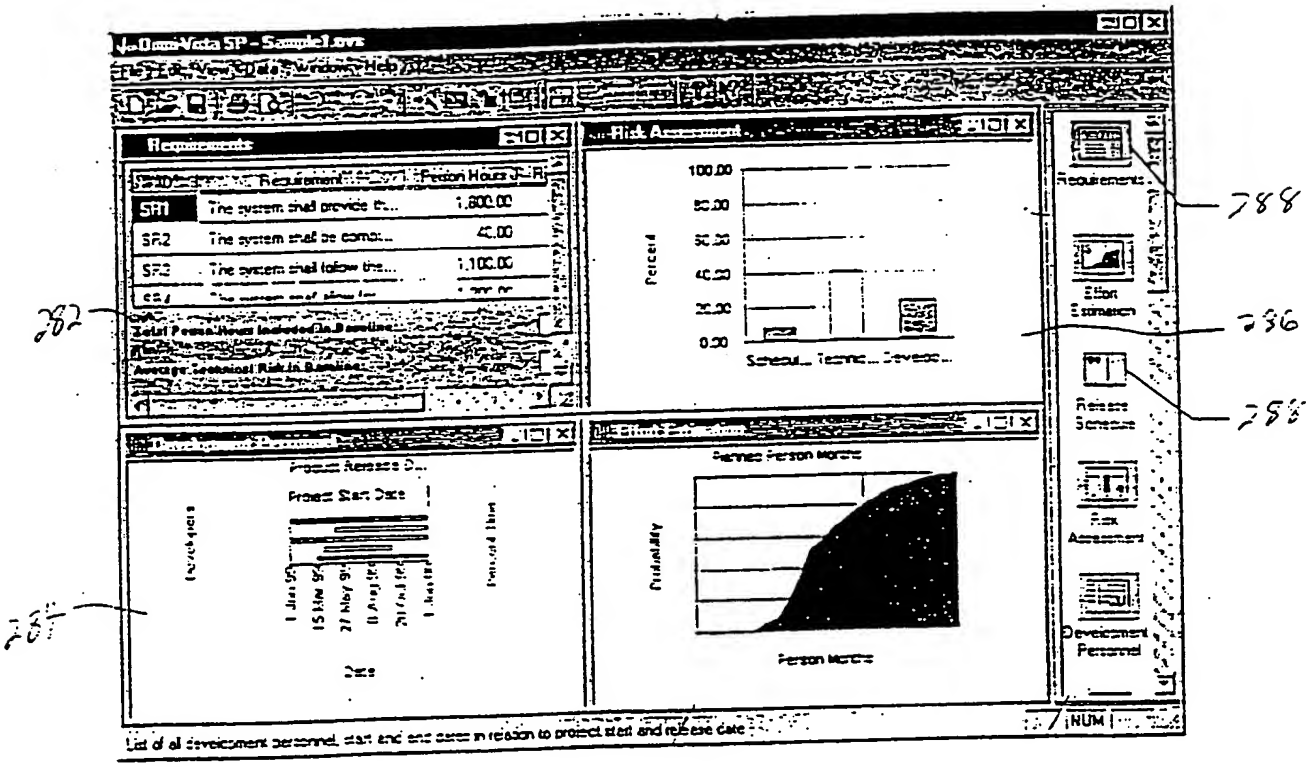

282
284
286
288
287
Requirements
Risk Assessment
Release Schedule
Development Personnel
Person Hours
Percent
Probability
Person Months
Date

FIG 15

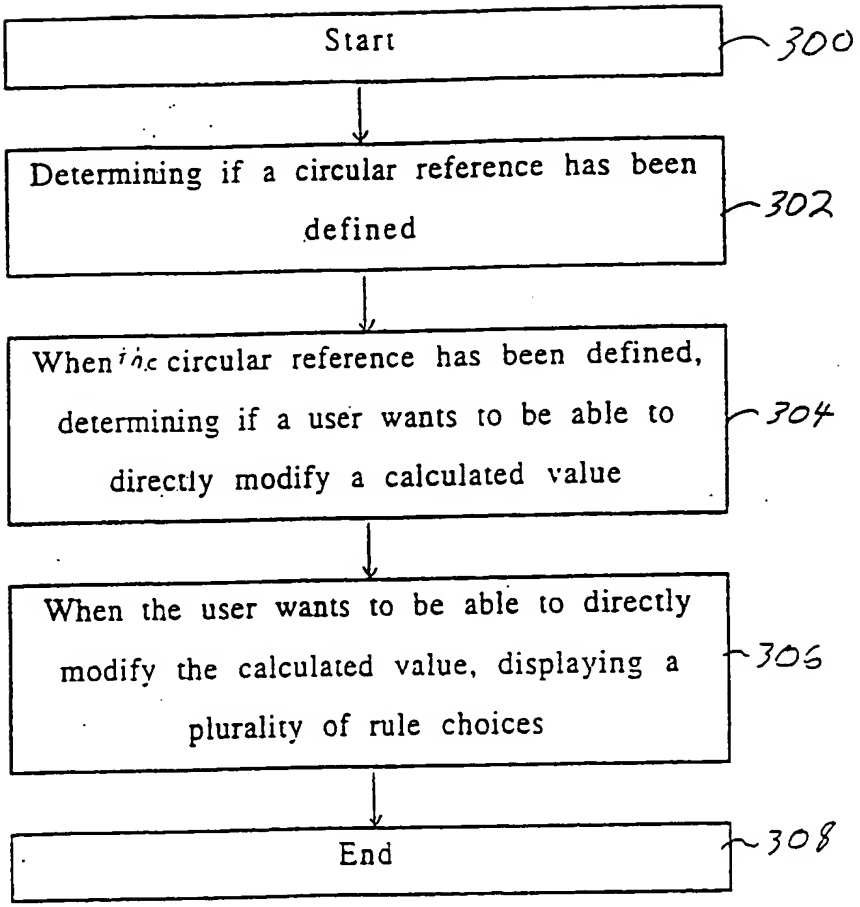
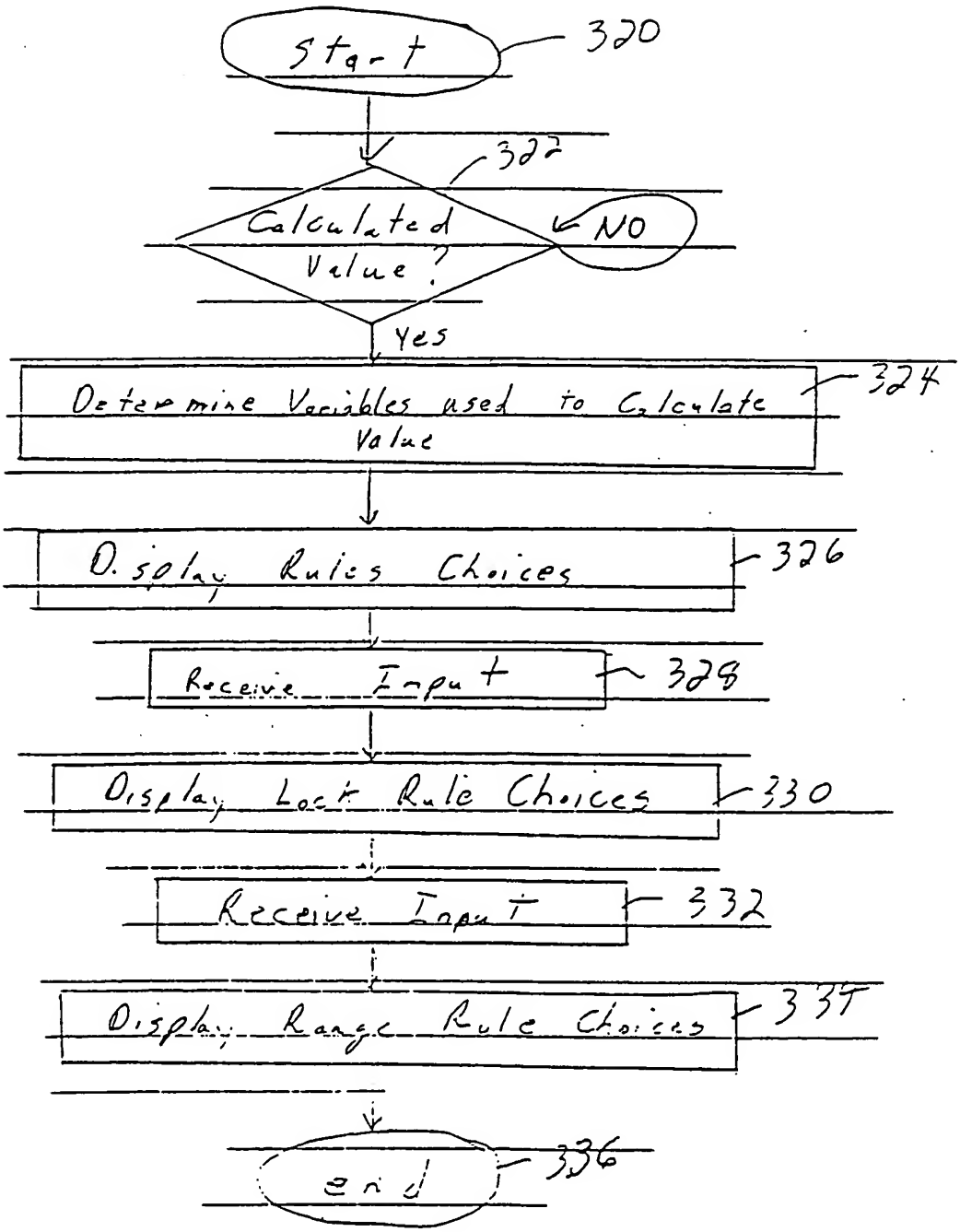

Start
300
Determining if a circular reference has been defined
302
When the circular reference has been defined, determining if a user wants to be able to directly modify a calculated value
304
When the user wants to be able to directly modify the calculated value, displaying a plurality of rule choices
306
End
308

FIG. 16


Start
320
322
Calculated Value ?
NO
Yes
Determine Variables used to Calculate Value
324
Display Rules Choices
326
Receive Input
328
Display Lock Rule Choices
330
Receive Input
332
Display Range Rule Choices
334
end
336

(1) $F = A + BC + D^E$

Sum Rule

$\Delta A = \boxed{a}\ \Delta F$

$\Delta(BC) = \boxed{b}\ \Delta F$

$\Delta(D^E) = \boxed{d}\ \Delta F$

where $a + b + d = 1$

If A Locked $a = 0$

Product Rule

$\Delta B = \boxed{f}(\Delta C)$

Exponential Rule

D - Locked

or

E - Locked

FIG. 18

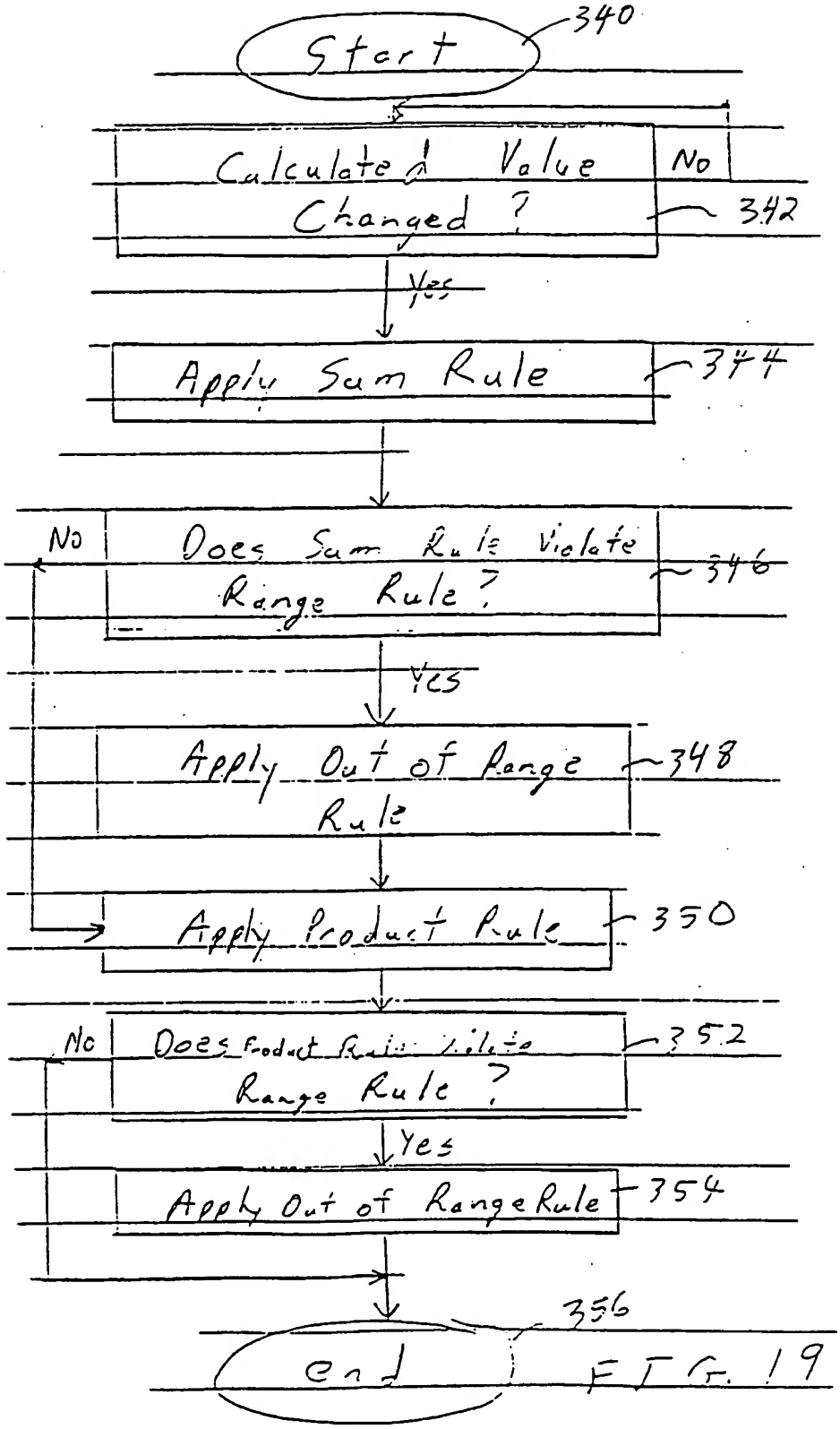

Start
340
Calculated Value Changed ?
No
342
Yes
Apply Sum Rule
344
No
Does Sum Rule Violate Range Rule ?
346
Yes
Apply Out of Range Rule
348
Apply Product Rule
350
No
Does Product Rule Violate Range Rule ?
352
Yes
Apply Out of Range Rule
354
End
356

FIG. 19

FIG. 20

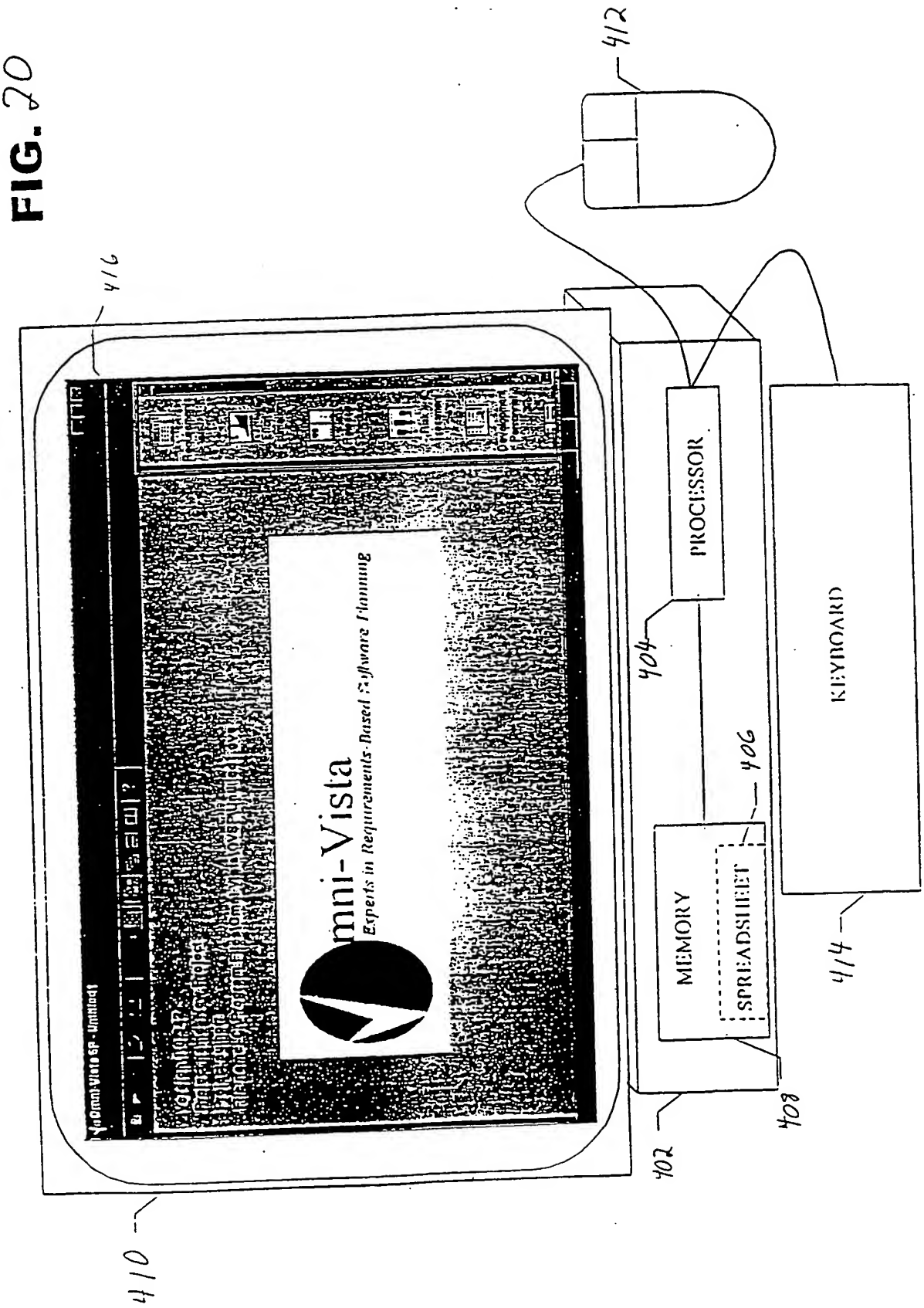
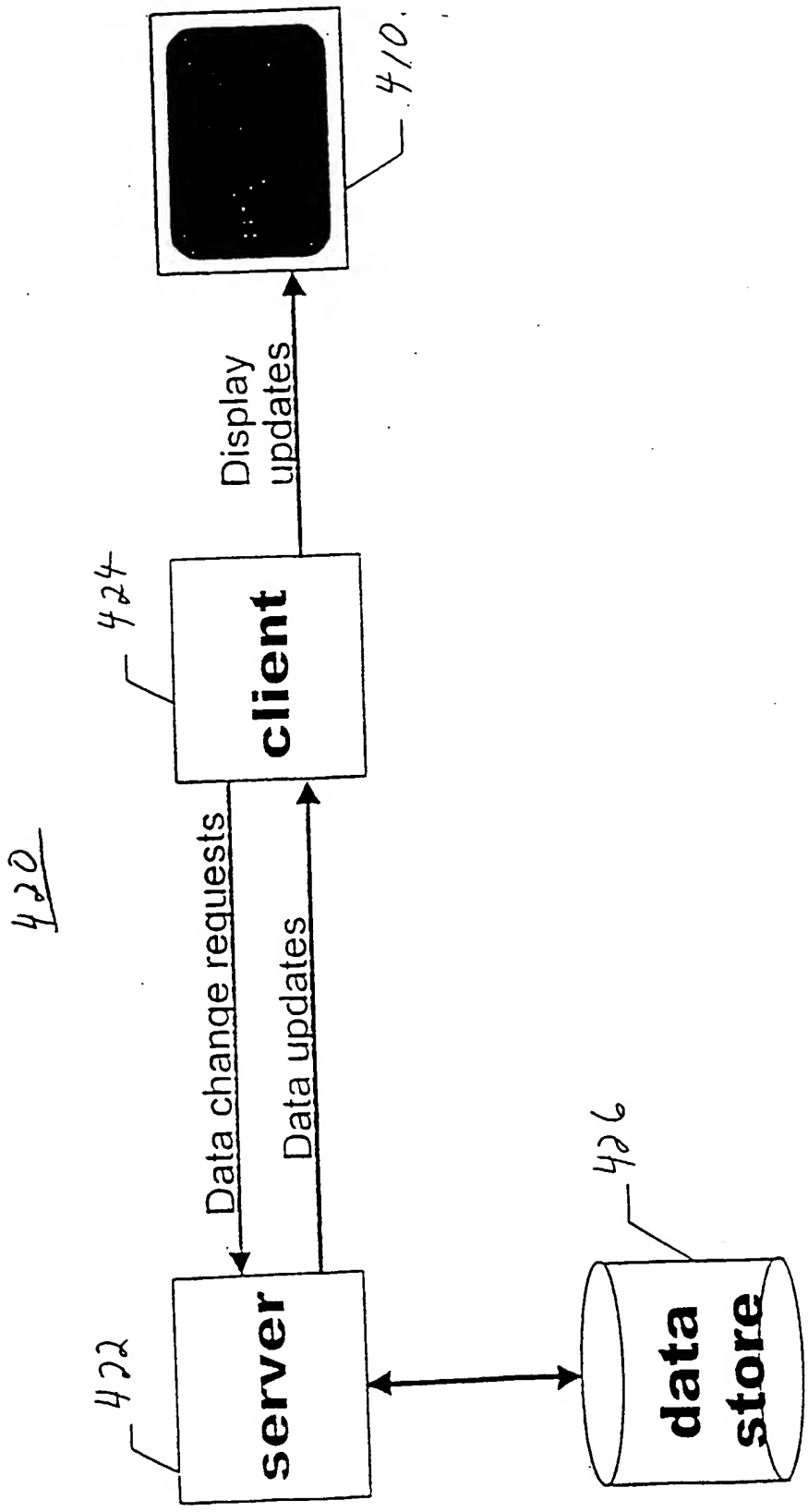

412
416
Omni-Vista
Experts in Requirements-Based Software Planning
PROCESSOR
MEMORY
SPREADSHEET
KEYBOARD
410
402
404
406
408
414

FIG. 21


410
Display updates
424
client
420
Data change requests
Data updates
426
422
server
data store

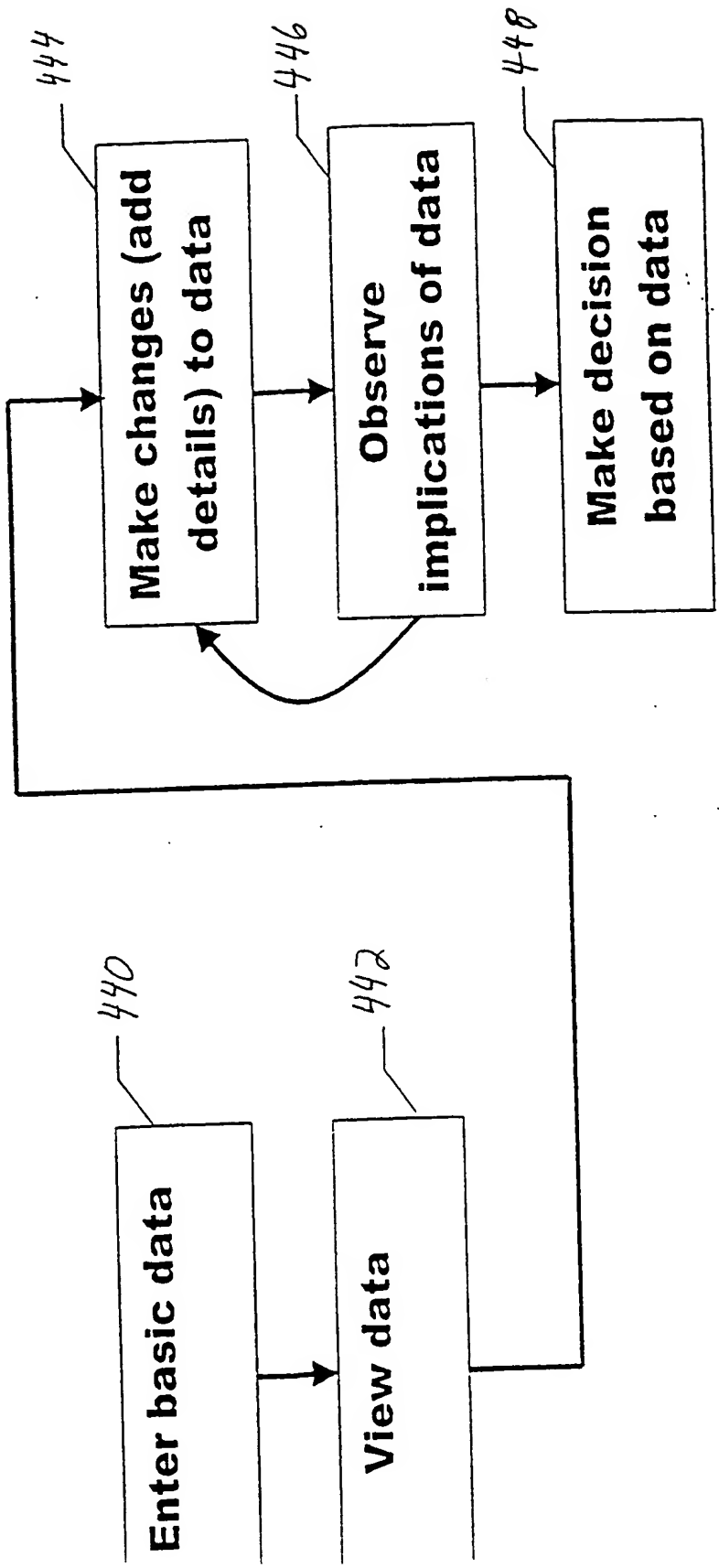

440
Enter basic data
442
View data
444
Make changes (add details) to data
446
Observe implications of data
448
Make decision based on data

FIG. 22

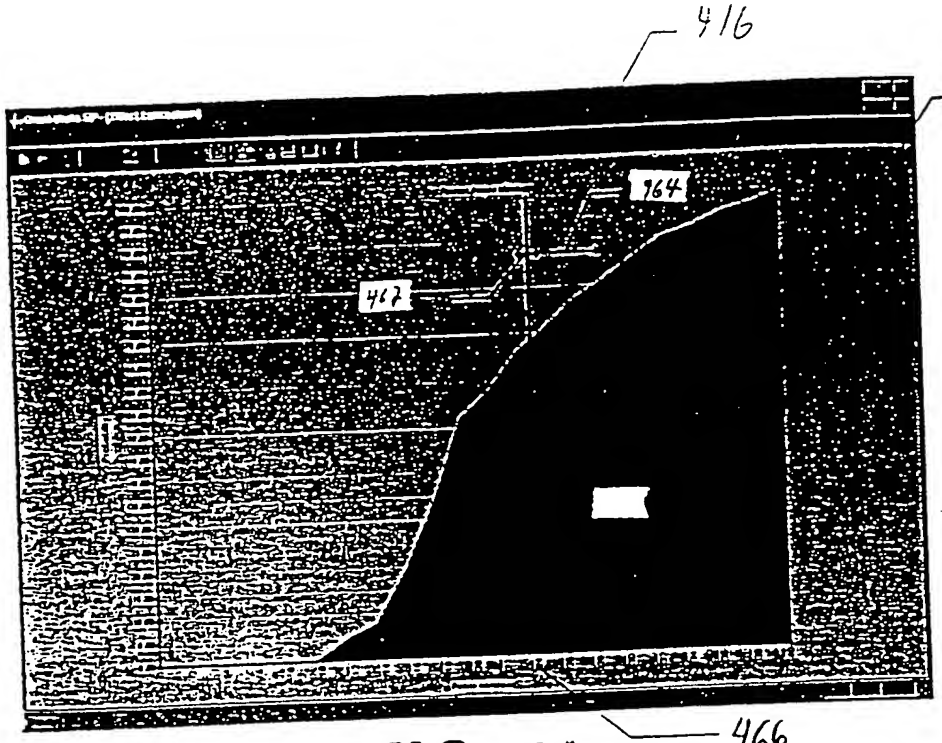


FIG. 23A

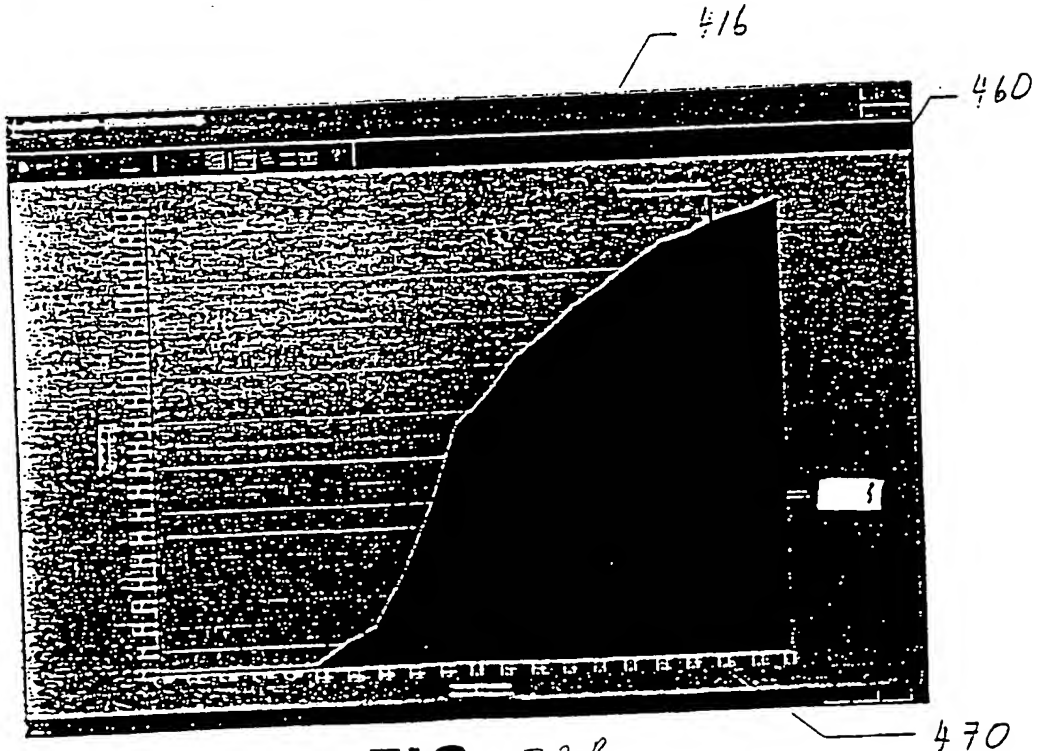


FIG. 23B

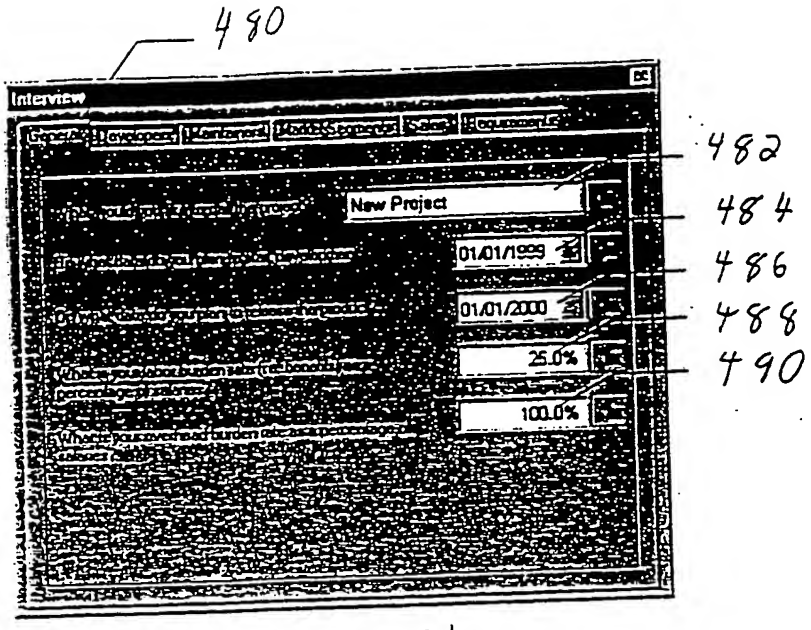

480
Interview
New Project
01/01/1999
01/01/2000
25.0%
100.0%
482
484
486
488
490

FIG. 24

FIG. 25

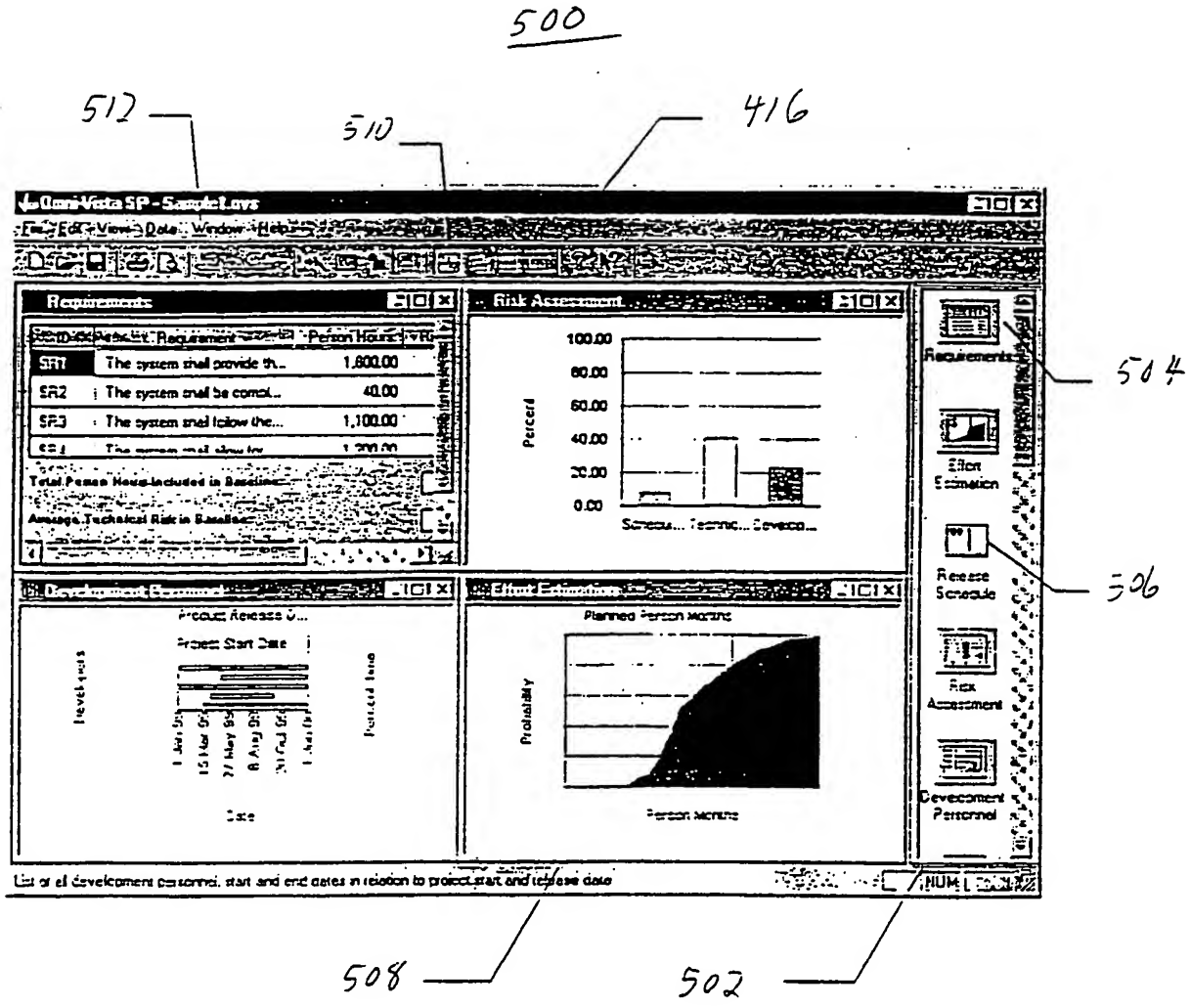
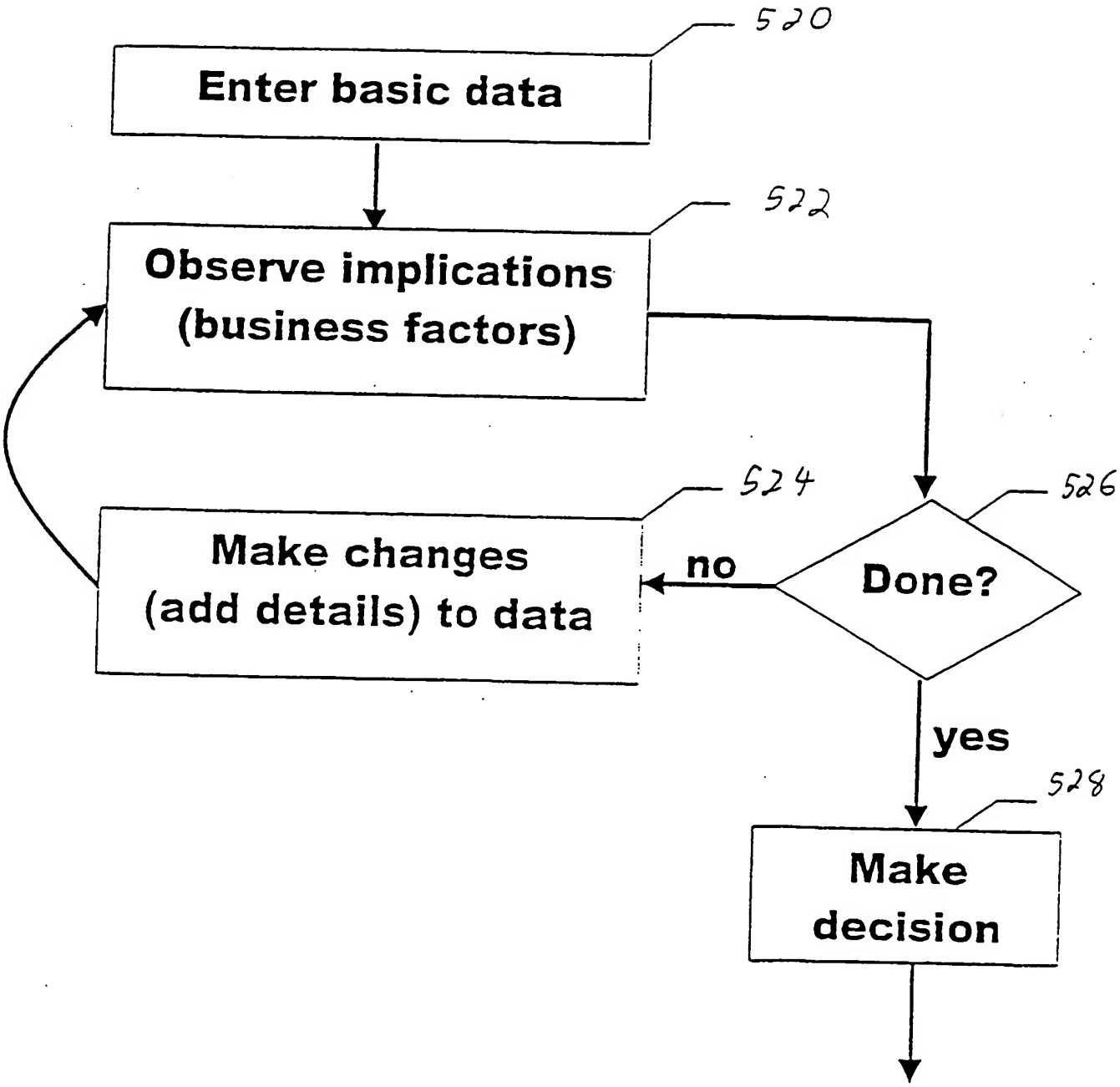

500
512
510
416
Requirements
Risk Assessment
Requirements
Effort Estimation
Release Schedule
Risk Assessment
Development Personnel
504
506
508
502

FIG. 26


520
Enter basic data
522
Observe implications (business factors)
524
Make changes (add details) to data
no
526
Done?
yes
528
Make decision

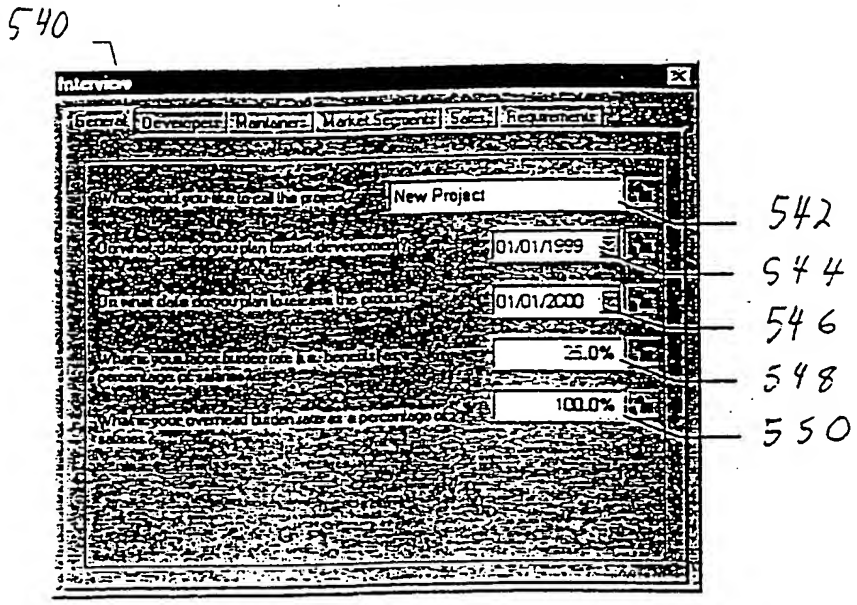

540
Interview
General
Developers
Marketers
Market Segments
Sales
Requirements
New Project
01/01/1999
01/01/2000
25.0%
100.0%
542
544
546
548
550

FIG. 5A

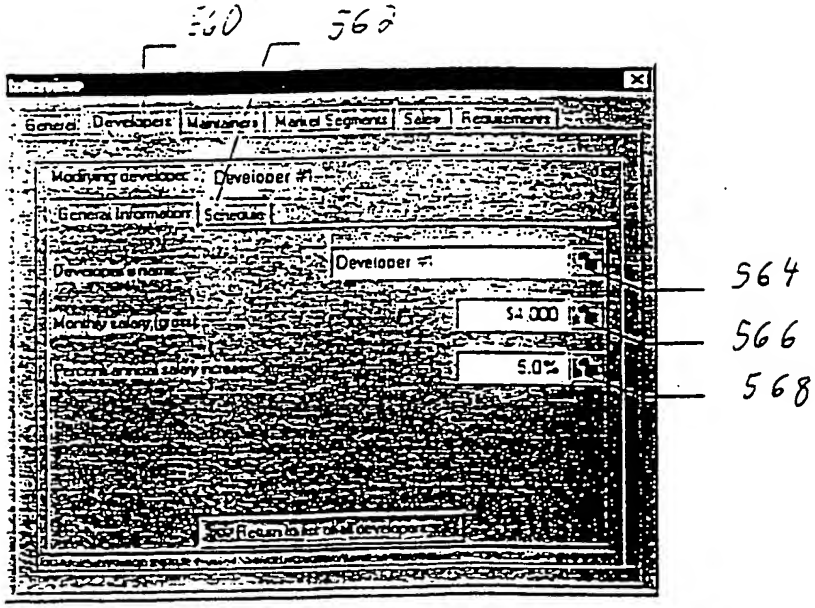

560
562
General
Developers
Marketers
Market Segments
Sales
Requirements
Modifying developer:
Developer #1
General Information
Schedule
Developer #1
$4,000
5.0%
564
566
568

FIG. 5B

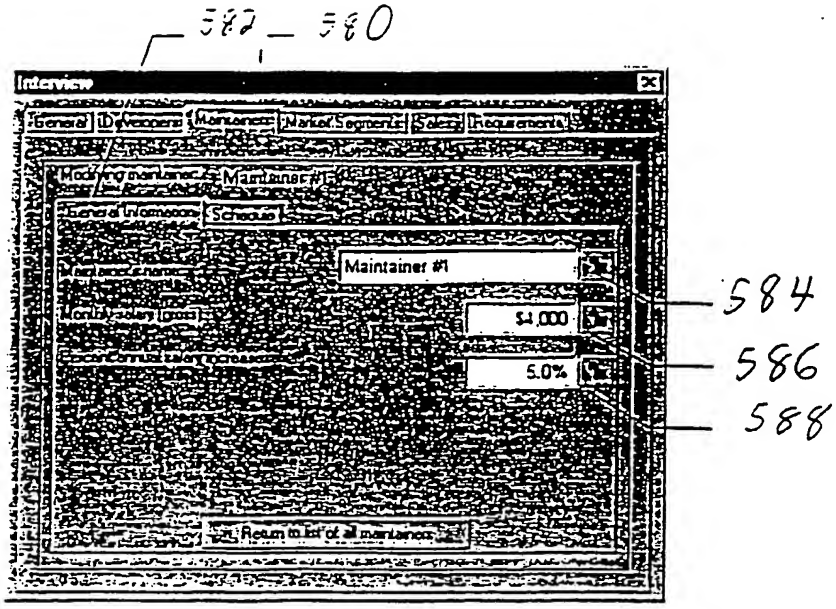


FIG. 27C

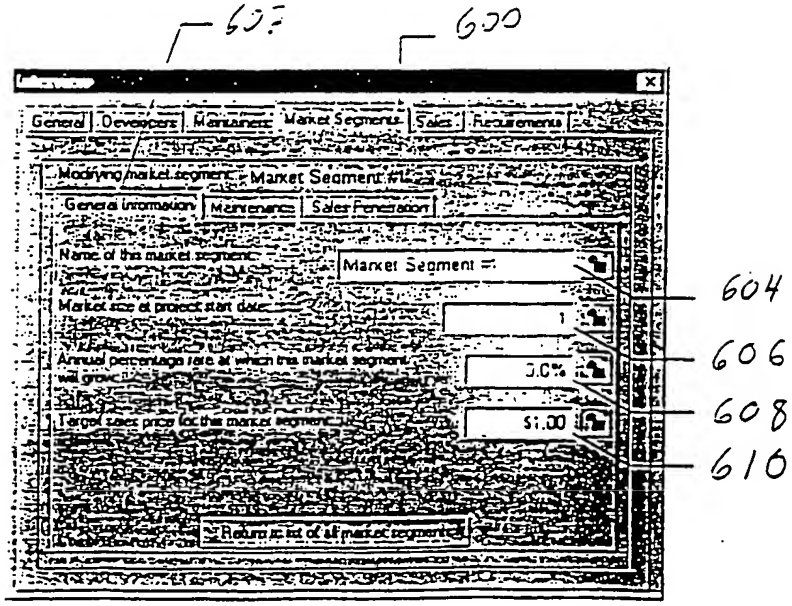


FIG. 27D

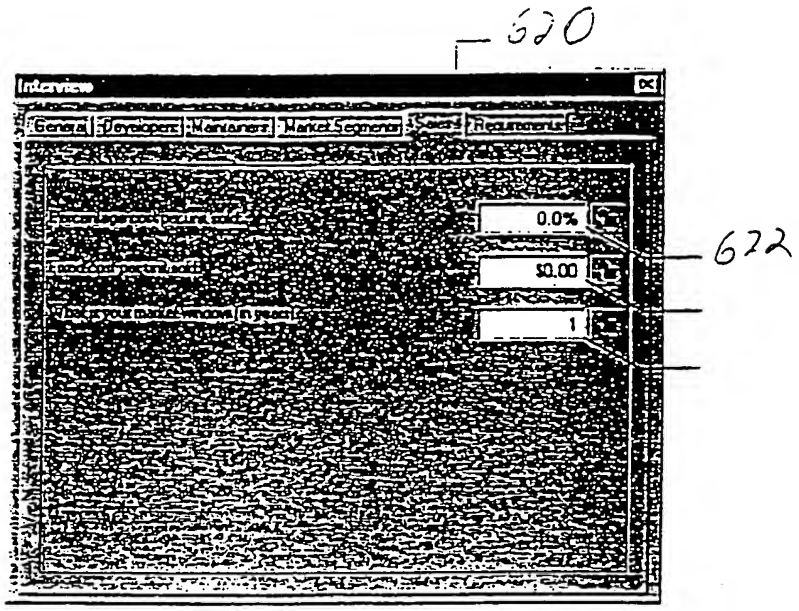


FIG. 27E

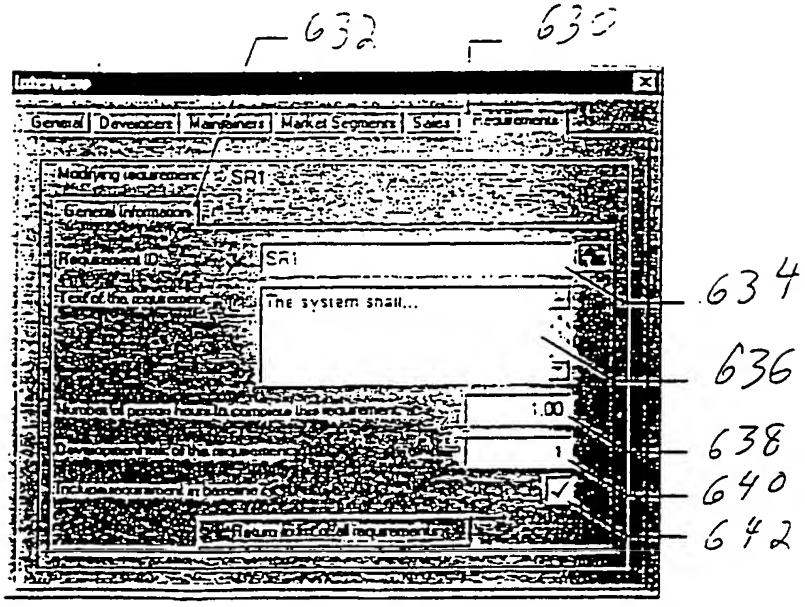
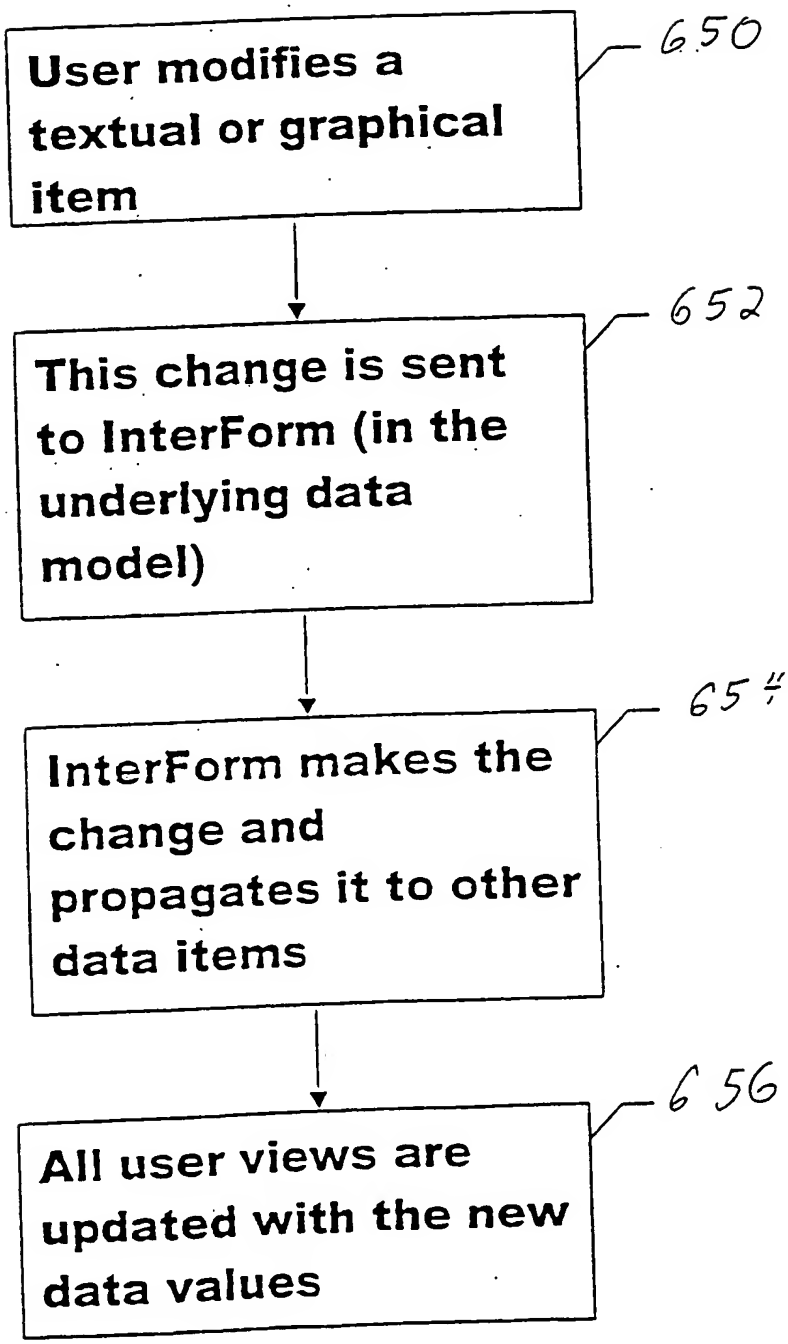


FIG. 27F

FIG. 28


User modifies a textual or graphical item
650
This change is sent to InterForm (in the underlying data model)
652
InterForm makes the change and propagates it to other data items
654
All user views are updated with the new data values
656

G. 29

660

(b)	(c)	(d)	(e)	(f)	(g)	(h)	(i)	(j)	(k)	(l)	(m)	(n)	(o)	(p)	(q)	(r)	(s)	(t)	(u)	(v)	
[redacted]						InterForm	Requirements	16	3	1300	1	0	0		0	FALSE					662
					1	InterForm	Requirements	15	3	0	0	1	0		FALSE						664
					1	Excel	Wizard	12	1			1	0								666
					1	Excel	Wizard	14	1			1	0								668
					1	Excel_CheckLock	Schedule	2	3			1	0		FALSE						670
					1	Excel	Schedule	2:10	4			1	0								672
					1	Excel_CheckLock	Cost	2	3			1	0		FALSE						674
					1	Excel	Cost	2:10	4			1	0								676
					1	Excel_CheckLock	Risk	2	2			1	0		FALSE						678
					1	Excel_CheckLock	Risk	2	4			1	0		FALSE						680
					1	Excel_CheckLock	Requirements	2	2			1	0		FALSE						682
					1	Excel_CheckLock	Requirements	6	1			1	0		FALSE						684

n Requested to be changed
quested Sheet
quested Min Row
quested Max Row
quested Min Column
quested Max Column
quested New Value / Priority
o Makes Change
licated Sheet
licated Row
licated Column

(l) New Value
(m) Old Value
(n) Runtime Row Offset
(o) Runtime Column Offset
(p) Range Iteration Offset
(q) Row Lock Status
(r) Priority Lock Status
(s) Cascade Status
(t) Cascade Value
(u) Error
(v) Comment

FIG. 30

690

(a)	(b)	(c)	(d)	(e)	(f)	(g)	(h)	(i)	(j)	(k)	(l)	(m)	(n)	(o)	(p)	(q)	(r)	(s)	(t)	(u)	(v)	
[redacted]							InterForm	Wizard	20	1	28.30	28.30	0	0								692
						1	InterForm	DevPerson	7	4	0.00%		0	0		FALSE	FALSE					694
						1	InterForm	DevPerson	0:9	7	Mar-98	Mar-98	0	0		FALSE						696
						1	Excel	Cost	2	2	28.30		0	0								698
						1	InterForm_Recalc	DevPerson	7	4	0.00%		0	0		FALSE		FALSE	FALSE			700
						2	InterForm	DevPerson	7	4	0.00%		0	0		FALSE	FALSE					702
						2	Excel	Cost	2	2	28.30		0	0								704
						2	InterForm_Recalc	Wizard	7	4	0.00%		0	0								706

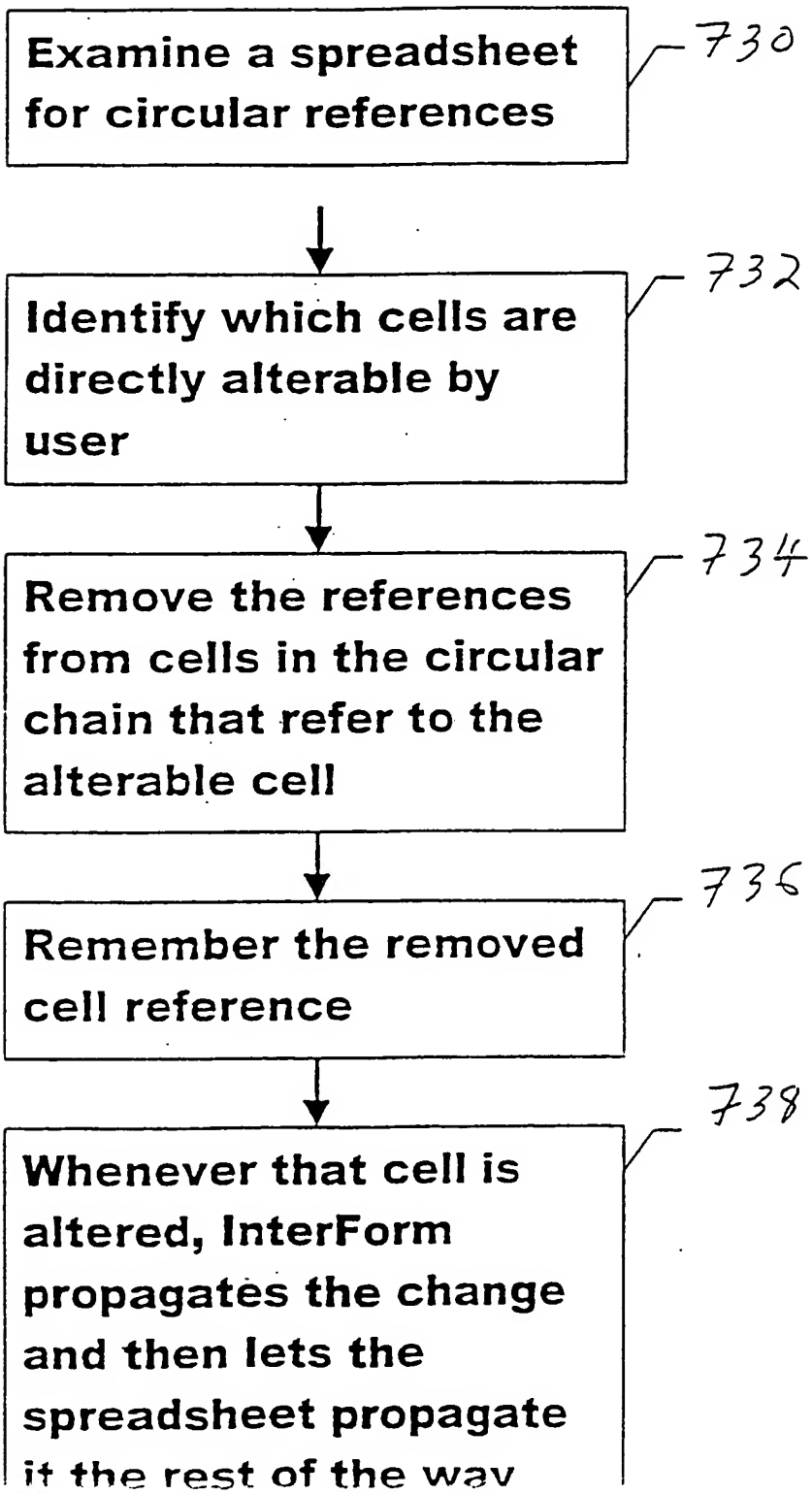
Item Requested to be changed
Requested Sheet
Requested Min Row
Requested Max Row
Requested Min Column
Requested Max Column
Requested New Value / Priority
Who Makes Change
Implicated Sheet
Implicated Row
Implicated Column

(l) New Value
(m) Old Value
(n) Runtime Row Offset
(o) Runtime Column Offset
(p) Range Iteration Offset
(q) Row Lock Status
(r) Priority Lock Status
(s) Cascade Status
(t) Cascade Value
(u) Error
(v) Comment

FIG. 31

```
                                                    710
                                                    |
blic Function InterForm (strRequestedSheet As String,  ——— 712
        nRequestedRow As Integer,  ——— 714
        nRequestedCol As Integer,  ——— 716
        nRequestedNewValue As Variant,  ——— 718
        bDoImplicated As Boolean,  ——— 720
        bDoAppCalc As Boolean)  ——— 722
        As Variant  ——— 724
```

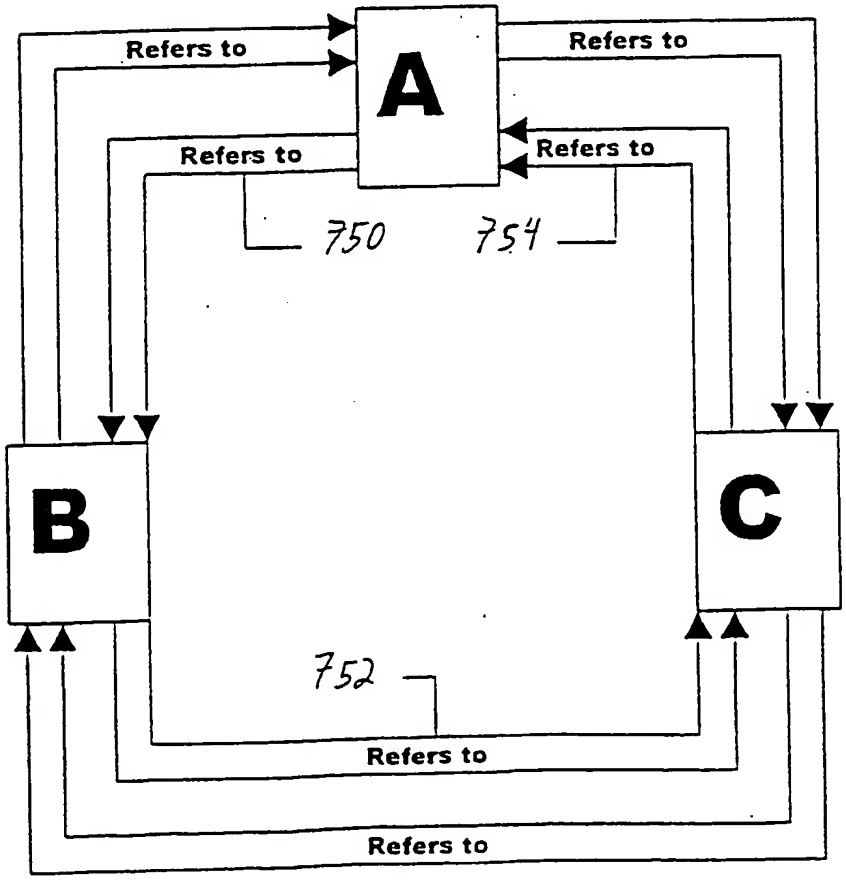
FIG. 32


Examine a spreadsheet for circular references
730
Identify which cells are directly alterable by user
732
Remove the references from cells in the circular chain that refer to the alterable cell
734
Remember the removed cell reference
736
Whenever that cell is altered, InterForm propagates the change and then lets the spreadsheet propagate it the rest of the way
738

740 742

	Cell Letter	Example	Formula of cell	Example Value
744	A	Cost	−B + C	2
746	B	Profit	C − A	4
748	C	Price	B + A	6

FIG. 33A



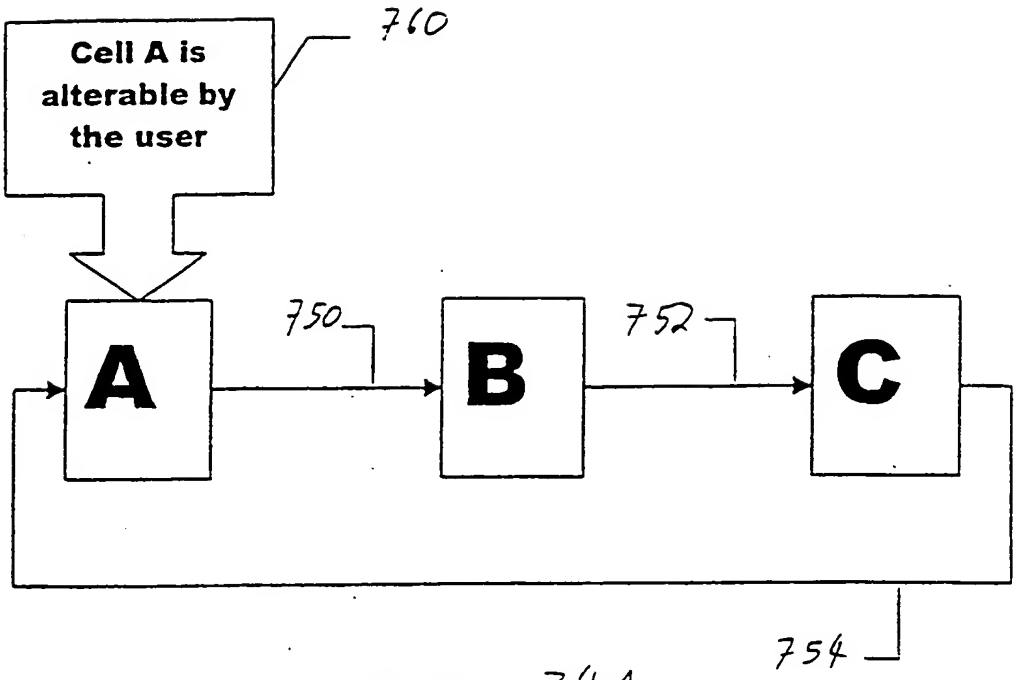


FIG. 34A

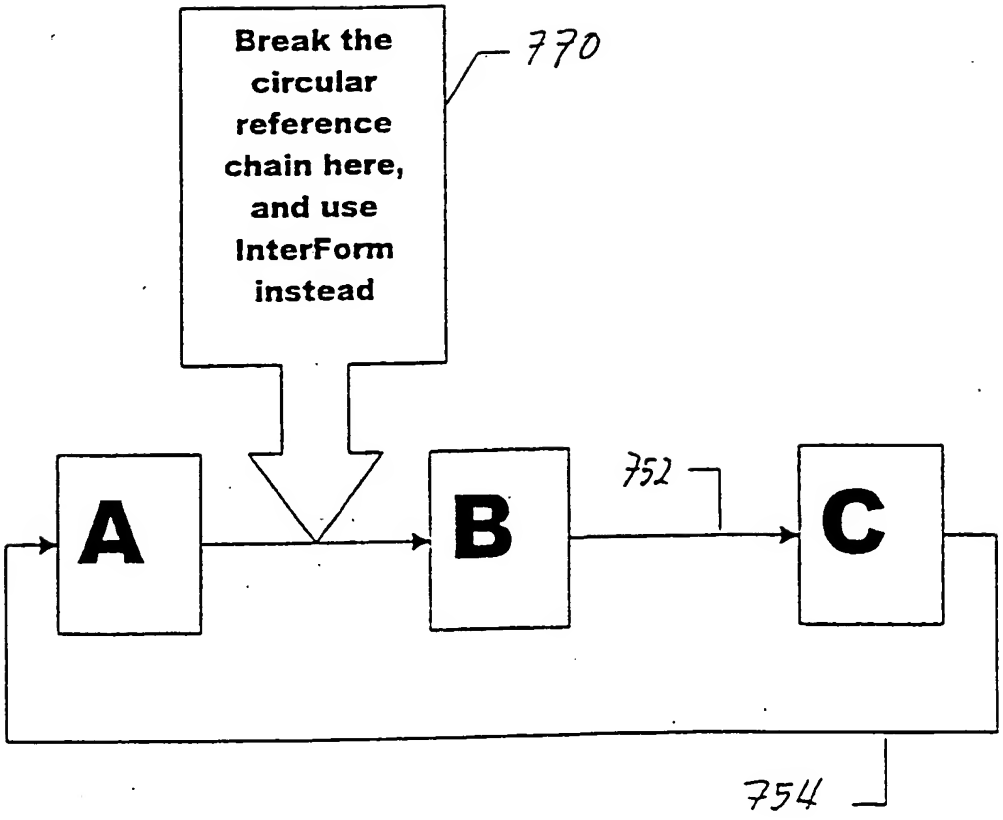

Break the circular reference chain here, and use InterForm instead
770
A
B
752
C
754

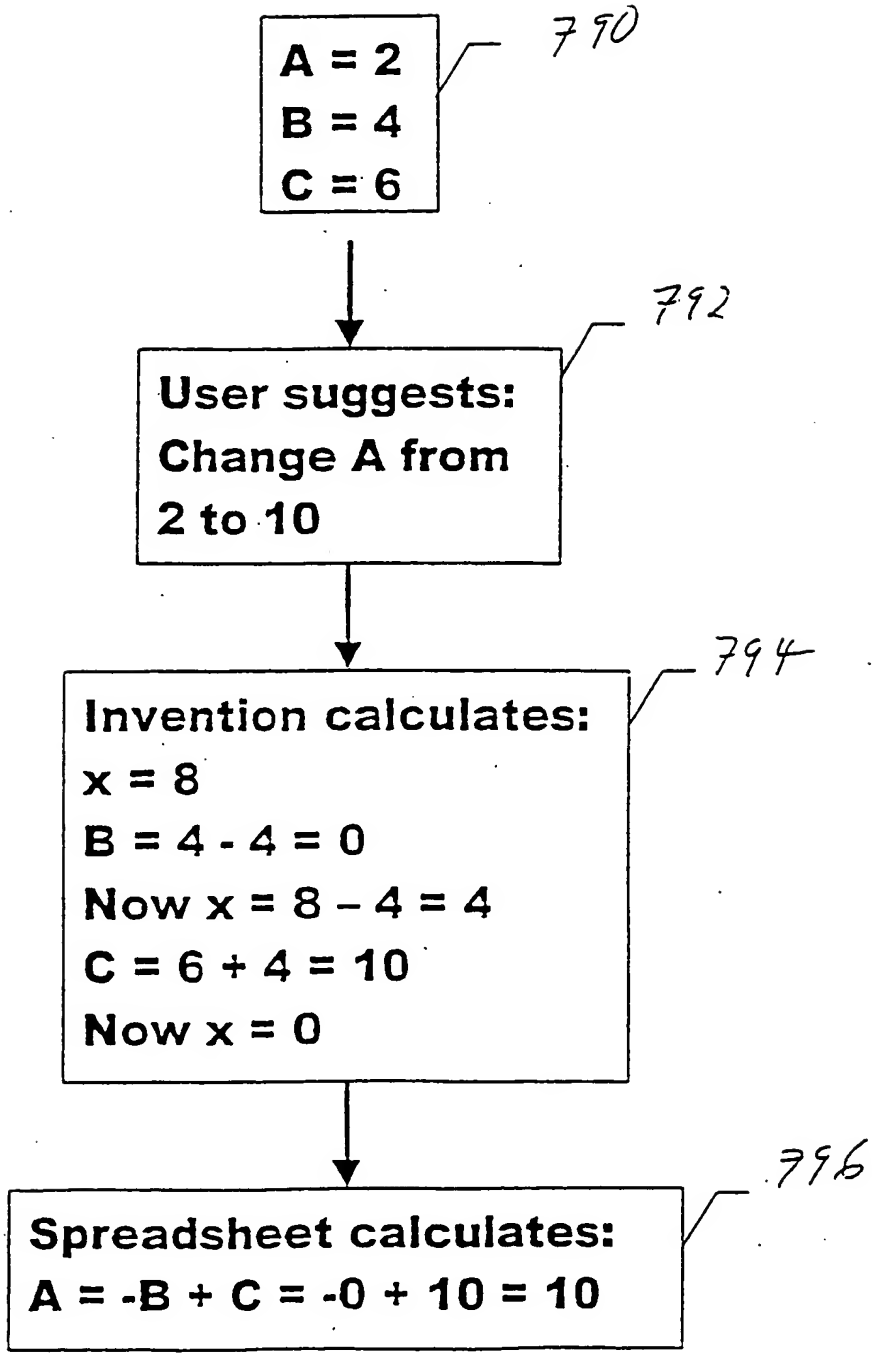
FIG. 34B

FIG. 35

740 742 780

Cell Letter	Original Formula of cell	New, non-circular formula
A	−B + C	−B + C
B	C − A	If user suggests a positive change to A, then let x = the difference between the original value of A and the suggested new value of A. Subtract as much of x as possible from B (do not allow B to go below the value of 0), and add the remainder of x to C. 782 784 If user suggests a negative change to A (do not allow A to go below the value of 0), then let x = the difference between the original value of A and the suggested new value of A. Add that change amount to B (and do nothing to C).
C	B + A	

FIG. 36


790
A = 2
B = 4
C = 6
792
User suggests:
Change A from
2 to 10
794
Invention calculates:
x = 8
B = 4 - 4 = 0
Now x = 8 – 4 = 4
C = 6 + 4 = 10
Now x = 0
796
Spreadsheet calculates:
A = -B + C = -0 + 10 = 10

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US99/16654

A. CLASSIFICATION OF SUBJECT MATTER

IPC(6) :G06F 17/17, 17/27
US CL : 707/503

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

U.S. : 707/503, 500, 104

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

APS, DIALOG
search terms: circular reference, spreadsheet, equation, graph

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A, P	US 5,796,932 A (FOX et al) 18 August 1998	1-8
A	US 5,652,880 A (SEAGRAVES) 29 July 1997	1-8
A	US 5,276,607 A (HARRIS et al) 04 January 1994	1-8
A	US 5,745,712 A (TURPIN et al) 28 April 1998	1-8
A	US 5,255,363 A (SEYLER) 19 October 1993	1-8

☐ Further documents are listed in the continuation of Box C. ☐ See patent family annex.

*	Special categories of cited documents:	"T"	later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
"A"	document defining the general state of the art which is not considered to be of particular relevance	"X"	document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
"E"	earlier document published on or after the international filing date	"Y"	document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art
"L"	document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)	"&"	document member of the same patent family
"O"	document referring to an oral disclosure, use, exhibition or other means		
"P"	document published prior to the international filing date but later than the priority date claimed		

Date of the actual completion of the international search 10 SEPTEMBER 1999	Date of mailing of the international search report 29 OCT 1999
Name and mailing address of the ISA/US Commissioner of Patents and Trademarks	Authorized officer